

JANEIRO DE 2025 (1º SEMESTRE)



ESCOLA SUPERIOR DE TECNOLOGIA
E GESTÃO DE ÁGUEDA

CultivUA

PROJETO TEMÁTICO EM
DESENVOLVIMENTO WEB

AUTORES

Daniel Silva | 113900

Gabriel Cravo | 87689

Lucas Duarte | 109190

Miguel Pirré | 114017

JANEIRO DE 2025 (1º SEMESTRE)



ESCOLA SUPERIOR DE TECNOLOGIA
E GESTÃO DE ÁGUEDA

CultivUA

PROJETO TEMÁTICO EM
DESENVOLVIMENTO WEB

PROFESSOR ORIENTADOR

Fábio Marques

AUTORES

Daniel Silva | 113900

Gabriel Cravo | 87689

Lucas Duarte | 109190

Miguel Pirré | 114017

Índice

1.	Introdução.....	1
2.	Enquadramento.....	3
3.	Objetivos do Projeto.....	5
3.1.	Promoção da Agricultura Urbana.....	5
3.2.	Sustentabilidade e Educação Ambiental	5
3.3.	Autossuficiência Alimentar e Consumo Consciente	5
3.4.	Bem-estar Físico e Mental	5
3.5.	Facilitar o Cuidado com as Plantas	5
3.6.	Melhorar a Experiência de Cultivo com Tecnologia	6
3.7.	Promover Comunidades Verdes e Cidades Sustentáveis	6
4.	Planeamento	7
4.1.	Cronograma e Recursos	7
4.1.1.	Calendarização das tarefas prevista	8
4.1.2.	Calendarização das tarefas executadas	9
4.1.3.	Análise do Cronograma: Planeamento e Execução do Projeto.....	10
4.2.	Metodologia	10
4.3.	Análise de Tecnologias	11
4.3.1.	<i>Frontend</i> : Angular	11
4.3.2.	<i>Backend</i> : Laravel	12
4.3.3.	Base de Dados: MySQL	13
4.3.4.	APIs Externas: Plant.id e Perennial	15
4.3.5.	Template Spike Admin Angular	15
4.3.6.	Stripe.....	16
4.3.7.	Bump.sh.....	17
4.4.	Divisão de Tarefas	18
5.	Levantamento do Estado de Arte.....	20
5.1.	FarmBot	20
	Funcionalidades Principais	20
	Pontos Fortes	20
	Limitações	21
	O que o CultivUA Pode Oferecer.....	21
5.2.	Click & Grow.....	22
	Funcionalidades Principais	22

Pontos Fortes	23
Limitações	23
O que o CultivUA Pode Oferecer.....	23
5.3. Click & Grow – Loja Online.....	24
Funcionalidades da Loja.....	24
Pontos Fortes	24
O que o CultivUA Pode Oferecer na Loja Online.....	24
5.4. Gardena Smart System.....	25
Funcionalidades Principais	25
Pontos Fortes	26
Limitações	26
O que o CultivUA Pode Oferecer.....	26
6. Requisitos Funcionais.....	29
6.1. Loja Online	29
Navegação na Loja	29
Compra de Produtos	29
Gestão de Stock	29
6.2. Gestão de Perfis de Utilizador.....	29
Registo de Utilizadores.....	29
Autenticação e Gestão de Conta.....	30
6.3. Registo e Gestão de Plantas	30
Registo de Plantas.....	30
Visualização do Histórico de Plantas	30
6.4. Dashboard Interativa	30
Monitorização em Tempo Real	30
Notificações e Alertas.....	31
Análise de Crescimento	31
6.5. Sensores IoT e Arduino.....	31
Integração com Sensores	31
Atualização Automática de Dados	31
Monitorização Remota	31
6.6. Integração com APIs Externas	32
Reconhecimento e Informação de Plantas.....	32
6.7. Gestão de Utilizadores Administradores.....	32
Gestão de Conteúdos	32
6.8. Relatórios e Estatísticas	32

Relatórios Personalizados.....	32
Estatísticas de Utilização	32
7. Requisitos Não Funcionais	33
7.1. Performance	33
Tempo de Resposta.....	33
Escalabilidade.....	33
7.2. Segurança.....	33
Proteção de Dados	33
Autenticação e Acesso.....	33
7.3. Usabilidade.....	34
Interface Intuitiva	34
Compatibilidade de Dispositivos.....	34
7.4. Manutenção.....	34
Atualizações Frequentes	34
Monitorização e Logs	34
8. Requisitos tecnológicos e de alojamento.....	35
8.1. Requisitos Tecnológicos	35
8.2. Desenvolvimento Local	35
8.3. Alojamento.....	36
9. Identificação dos Utilizadores	37
9.1. Cidadãos Urbanos e Suburbanos Interessados em Agricultura.....	37
9.2. Entusiastas de Tecnologia e Agricultura.....	37
9.3. Administradores da Plataforma.....	38
9.4. Utilizadores com Necessidades Especiais.....	38
9.5. Persona 1	39
9.6. Persona 2.....	40
10. Casos de utilização.	42
10.1. Loja Online	42
10.2. Gestão de Utilizadores	44
10.3. Monitorização e Gestão de Plantas	45
10.4. Sensores	46
11. Modelos de base de dados	48
11.1. Modelo Conceptual	48
11.2. Modelo Relacional	50
11.3. Otimização da Base de Dados com Índices	52
Índices Implementados	52

11.4.	Automatização do Sistema com Triggers	53
	Triggers Implementados	53
12.	Arquiteturas de Informação	56
12.1.	Objetivos da Arquitetura de Informação	56
12.2.	Estruturas de Informação	56
	Dashboard de Administrador	56
	<i>Dashboard</i> de Utilizador	57
	Loja Online	58
	Quiz	58
	Blog	58
	Página Inicial	59
12.3.	User Flow	59
13.	Diagrama MVC	60
14.	Diagrama de Arquitetura do Sistema	62
15.	Estratégia para Promover a Acessibilidade da Aplicação	63
15.1.	Princípios Gerais de Acessibilidade	63
15.2.	Funcionalidades Específicas de Acessibilidade	63
15.3.	Testes de Acessibilidade	64
15.4.	Usabilidade e Responsividade	64
16.	Wireframes	65
16.1.	Landing Page	66
16.2.	Loja Online	67
17.	Protótipo de Alta-fidelidade	68
17.1.	Página Inicial (Landing Page)	68
17.2.	Dashboard do Admin	70
18.	Demonstração do Website	72
18.1.	Página Inicial (Landing Page)	72
18.2.	Dashboard do Admin	74
18.3.	Produtos	75
18.4.	Categorias	76
18.5.	Blog Admin	77
18.6.	Tickets	78
18.7.	<i>Dashboard</i> do Utilizador	79
18.8.	Detalhes da Planta	80
18.9.	Loja Online	81
	Checkout	82

18.10.	Loja Online – Compra de Kit	83
18.11.	Gestão da Conta	84
18.12.	Contactos	85
18.13.	Quiz	86
18.14.	Identificar Planta.....	87
18.15.	Chatbot – Plant Lover.....	88
19.	Testes e Validação.....	89
19.1.	Testes de Sistema e Validação	89
19.2.	Testes Unitários.....	90
19.2.1.	Testes Unitários no Angular.....	91
19.2.1.	Testes Unitários em Laravel.....	93
19.3.	Testes de qualidade com o Lighthouse	95
19.4.	Testes de usabilidade	96
19.4.1.	Tabela de Tarefas Utilizadas nos Testes de Usabilidade.....	97
20.	Limitações e Problemas Encontrados.....	99
20.1.	Problemas Encontrados nos Testes de Qualidade e Ações de Melhoria.....	99
20.2.	Resultados dos Testes de Usabilidade e Ações de Melhoria	101
20.2.1.	Problemas Identificados	101
20.2.2.	Ações de Melhoria.....	102
20.3.	Problemas Identificados Através do Feedback do Professor	103
	Problema de legibilidade causado pelo uso de verde-claro nos textos.....	103
	Informações desnecessárias nas tabelas da Dashboard admin	104
	Problema de ausência de simbologia no preço.....	104
20.4.	Limitações no projeto.....	105
	Problema de Limitação na API de Identificação de Plantas.....	105
21.	Análise de resultados.....	107
22.	Reflexão Crítica e Conclusões.....	109
22.1.	Sugestões para o futuro	109
22.2.	Síntese das Experiências	110
23.	Conclusão	111
24.	Bibliografia.....	112
Anexos.....		113
Anexo A		114
ANEXO B	Erro! Marcador não definido.	
Anexo B.....		115
Anexo C.....		116

Loja Online.....	116
1. Adicionar Produto ao Carrinho.....	116
2. Remover Produto do Carrinho.....	116
3. Visualizar o Carrinho	117
4. Checkout.....	117
5. Pagamento de Produtos	118
6. Comprar Produtos na Loja.....	118
Gestão de Utilizadores.....	119
7. Registar-se	119
8. Autenticar	120
9. Editar Perfil	120
10. Consultar Histórico de Compras.....	121
Monitorização e Gestão de Plantas.....	121
11. Adicionar Nova Planta	121
12. Consultar Informações de Cuidados.....	122
13. Visualizar Detalhes e Características da Planta.....	122
14. Visualização do Histórico de Plantas --.....	123
15. Regar Plantas com Base em Calendário	123
16. Monitorizar Plantas	124
17. Receber Notificações e Alertas --	124
18. Reconhecer Automaticamente Plantas	125
19. Consultar informações e cuidados.....	125
20. Associar kit.....	126
21. Criar ticket suporte	126
Blog e Conteúdos	127
22. Gerir Conteúdo do Blog	127
23. Comentar artigo do blog	127
24. Visualizar Artigos do Blog.....	128
Gestão de Dados e Relatórios.....	128
25. Gerir Categorias.....	128
26. Gerir Produtos	129
27. Gerir Localizações.....	129
28. Visualizar histórico de tickets.....	130
29. Gerar Relatório de Estatísticas	130
Notificações e Alertas.....	131
30. Receber Notificações e Alertas Administrativos	131

31. Receber Notificação Newsletter	132
Sensores.....	132
32. Enviar Notificações de Alerta para o Sistema	132
33. Monitorizar Condições da Planta	133
Outros Casos de Utilização	134
34. Participar no Quiz Personalizado	134
Anexo D	135
Anexo E.....	136
Anexo F.....	137
Anexo G	138
Anexo H	139
Anexo I.....	140
Anexo J.....	141
Anexo K	142
ANEXO L.....	143

Índice de Figuras

Figura 1 - Calendarização das tarefas prevista	8
Figura 2 - Calendarização das tarefas executadas	9
Figura 3 - Tabela de divisão de tarefas.....	Erro! Marcador não definido.
Figura 4 - FarmBot	22
Figura 5 - Click & Grow	25
Figura 6 - Gardena Smart System	27
Figura 7 - Gardena Produtos.....	27
Figura 8 - Modelo Conceptual da Base de Dados	49
Figura 9 - Modelo Relacional da Base de Dados	51
Figura 10 - Diagrama MVC	61
Figura 11 - Diagrama de Arquitetura do Sistema.....	62
Figura 12 - Landing Page	66
Figura 13 - Loja Online	67
Figura 14 - Mockup Landing Page	69
Figura 15 - Mockup Dashboard Admin.....	71
Figura 16 - Dashboard Landing Page	73
Figura 17 - Dashboard do Admin.....	74
Figura 18 - Dashboard Produtos Admin.....	75
Figura 19 - Dashboard Categorias Admin.....	76
Figura 20 - Dashboard Blog Admin	77
Figura 21 - Dashboard Tickets Admin	78
Figura 22 - Dashboard do Cliente	79
Figura 23 - Detalhes Planta (Características)	80
Figura 24 - Mockup Loja Online.....	81
Figura 25 - Checkout.....	82
Figura 26 - Checkout Finalizado	Erro! Marcador não definido.
Figura 27 - Perfil Utilizador	84
Figura 28 - Contactos	85
Figura 29 - Quiz	86
Figura 30 - Identificar planta.....	87
Figura 31 - ChatBot.....	88
Figura 32 - Teste unitário ao service do carrinho de compras.....	91
Figura 33 - Resultado do teste do service do carrinho de compras	92
Figura 34 - Teste unitário á componente login da plataforma.....	92

Figura 35 - Resultado dos testes da componente de login da plataforma.....	92
Figura 36 - Teste unitário ao controller responsável pela criação de Tickets.....	94
Figura 37 - Resultado do teste ao controller de contactos	95
Figura 38 - Resultado do teste de qualidade do lighthouse no desktop.....	95
Figura 39 - Resultado do teste de qualidade do lighthouse em mobile	96
Figura 40 - Gráfico de Requisitos Cumpridos Não Cumpridos Parcialmente Cumpridos	107

Agradecimentos

Este projeto simboliza a materialização de uma jornada desafiadora e é o reflexo do nosso esforço coletivo e da dedicação de todos os membros do grupo. No entanto, a sua concretização só foi possível graças ao apoio e contributo inestimável de pessoas que, direta ou indiretamente, se envolveram neste percurso.

Manifestamos a nossa mais profunda gratidão ao Professor Fábio José Reis Luís Marques, o nosso orientador, cuja sabedoria e orientação incansável foram cruciais para o sucesso deste projeto.

Dirigimos igualmente os nossos sinceros agradecimentos a todos os outros docentes do nosso curso. O apoio contínuo e os conhecimentos partilhados ao longo da nossa formação foram fundamentais, não só para a realização deste trabalho, mas também para o desenvolvimento da nossa capacidade metodológica e do nosso crescimento intelectual.

Agradecemos ainda à ESTGA e a todos aqueles que, de forma direta ou indireta, deixaram a sua marca na nossa trajetória. Um especial reconhecimento vai para todos os que participaram nos testes de usabilidade, cuja colaboração foi essencial para validar e aperfeiçoar a plataforma. O apoio, as palavras de incentivo e as aprendizagens proporcionadas ao longo desta caminhada foram pilares essenciais para o nosso desenvolvimento, tanto a nível pessoal como profissional.

Por fim, deixamos um sentido bem-haja a todos que fizeram parte desta importante etapa da nossa vida académica. Este projeto é também vosso, pelo impacto positivo que tiveram na sua realização.

1. Introdução

No âmbito do Projeto Temático em Desenvolvimento Web, integrado no primeiro semestre do terceiro ano da Licenciatura em Tecnologias da Informação da Escola Superior de Tecnologias e Gestão de Águeda, foi proposta aos alunos a criação de uma aplicação web que integrasse desenvolvimento *fullstack*. O objetivo deste projeto é oferecer uma experiência prática e completa, que aborde tanto o *frontend* quanto o *backend*, assim como a criação e manipulação de bases de dados. Em resposta a esse desafio, o grupo optou por desenvolver o **CultivUA**, uma plataforma inovadora voltada para a promoção da agricultura urbana e a sustentabilidade, através de uma interface web interativa e de fácil utilização.

Com o **CultivUA**, pretende-se explorar soluções inovadoras que incentivem a agricultura em espaços urbanos, proporcionando práticas sustentáveis e acessíveis a utilizadores com recursos limitados de espaço e tempo. Esta plataforma distingue-se pela capacidade de fornecer informações específicas sobre as necessidades das plantas, auxiliando na monitorização e nos cuidados necessários, de forma a tornar o cultivo urbano mais prático e acessível a todos.

Este relatório documenta todas as etapas e metodologias seguidas ao longo do desenvolvimento do projeto, incluindo o Planeamento e Execução das Tarefas, com as divisões específicas de trabalho realizadas pela equipa; uma análise sobre o Estado da Arte, que contextualiza o **CultivUA** no panorama atual das aplicações voltadas para a agricultura urbana e sustentável; e os Casos de Utilização, que detalham os principais fluxos de interação dos utilizadores com a aplicação.

Em seguida, o relatório apresenta a Análise de Tecnologias que justifica a escolha de frameworks e ferramentas como Laravel[1] e Angular[2] para a implementação da plataforma. Inclui também uma descrição técnica dos Modelos de Dados e do Modelo Estrutural da Aplicação, bem como uma análise aprofundada das fases de Desenvolvimento da Aplicação, onde são detalhados o processo de construção, testes e validação do sistema.

Finalmente, o relatório conclui com uma Análise de Resultados e uma Reflexão Crítica sobre os objetivos alcançados e os desafios enfrentados durante o projeto, acompanhados da Conclusão que discute o impacto e as perspetivas futuras do **CultivUA**.

2. Enquadramento

A agricultura urbana tem emergido como uma solução sustentável para os desafios ambientais e sociais decorrentes do crescimento populacional e do consumo elevado de recursos naturais. Cultivar alimentos localmente reduz desperdícios, minimiza a dependência do transporte de bens alimentares e contribui para a diminuição das emissões de carbono. Neste contexto, o **CultivUA** apresenta-se como uma plataforma inovadora e prática, que promove a sustentabilidade urbana e capacita os cidadãos a cultivarem as suas próprias plantas e alimentos frescos para consumo próprio.

O **CultivUA** combina a utilização de tecnologias robustas e modernas, como o Laravel e o Angular, para criar uma plataforma escalável, eficiente e acessível. Esta solução oferece ferramentas e funcionalidades avançadas que permitem aos utilizadores monitorizar e controlar os principais parâmetros ambientais das suas plantas em tempo real, com o suporte de sensores IoT e uma interface de gestão intuitiva. Além disso, a plataforma incorpora recursos como identificação de plantas por imagem, quiz interativo, *chatbot*, registo histórico de cuidados e condições das plantas, bem como alertas personalizados.

Para além de promover a autossuficiência alimentar, o **CultivUA** visa reduzir a pegada ecológica, melhorar a qualidade de vida urbana e contribuir para a criação de cidades mais verdes e resilientes. A oferta é adaptada a diferentes níveis de experiência em jardinagem, de maneira a garantir uma experiência personalizada e inclusiva, ajustada ao estilo de vida e às necessidades de cada utilizador.

Plataforma Web

A plataforma web do **CultivUA** será o núcleo da solução e será desenvolvida com tecnologias modernas. No *back-end*, o Laravel assegurará a robustez e a escalabilidade necessárias, enquanto no *front-end*, o Angular garantirá uma interface dinâmica, responsiva e intuitiva para todos os utilizadores.

Loja Online

A loja online será uma porta de entrada fundamental para os utilizadores iniciarem a sua jornada na agricultura urbana, oferecendo uma vasta gama de produtos essenciais para o cultivo doméstico.

Entre estes:

- **Ferramentas de Cultivo:** Incluem sementes, substratos, vasos e ferramentas essenciais para iniciar na jardinagem.
- **Kits de Dispositivos IoT:** Sensores de luz, humidade, temperatura, entre outros dispositivos baseados em Arduino, integrados à plataforma, para monitorizar dados ambientais em tempo real.

Os produtos são organizados para atender perfis variados de utilizadores, desde iniciantes com pouco espaço disponível até jardineiros urbanos mais experientes, de forma a promover a democratização do acesso à agricultura urbana.

3. Objetivos do Projeto

Os principais objetivos do CultivUA são:

3.1. Promoção da Agricultura Urbana

- **Incentivar a produção de alimentos em ambientes urbanos:** Capacitar cidadãos que vivem em centros urbanos, com pouco ou nenhum espaço exterior, a cultivar alimentos e plantas ornamentais, promovendo a prática do "autocultivo" mesmo em espaços pequenos.

3.2. Sustentabilidade e Educação Ambiental

- **Fomentar práticas ecológicas:** Educar os utilizadores para a adoção de métodos sustentáveis de cultivo, como a redução de desperdícios e a utilização responsável de recursos.
- **Educar para o cultivo responsável:** Ensinar boas práticas de jardinagem e sensibilizar para a preservação ambiental através da plataforma.

3.3. Autossuficiência Alimentar e Consumo Consciente

- **Promover a produção e consumo de alimentos frescos:** Incentivar os cidadãos a cultivarem as suas próprias frutas, vegetais e ervas, fomentando hábitos alimentares mais saudáveis e conscientes.

3.4. Bem-estar Físico e Mental

- **Oferecer uma atividade relaxante e educativa:** Promover o cultivo de plantas como uma atividade que melhora o bem-estar físico e mental, ajudando a reduzir o stress e a aumentar o equilíbrio emocional.

3.5. Facilitar o Cuidado com as Plantas

- **Monitorização inteligente e notificações automáticas:** Através dos sensores IoT, a plataforma notifica os utilizadores automaticamente sobre as necessidades das suas plantas, como rega ou fertilização, com base em dados recolhidos em tempo real.

- **Sistemas automatizados para cuidados personalizados:** A plataforma oferece dicas específicas de cuidados para cada planta, enviando lembretes e sugestões no momento certo.

3.6. Melhorar a Experiência de Cultivo com Tecnologia

- **Gestão prática através da tecnologia:** A utilização de sensores e APIs permite uma monitorização eficiente e automatizada das plantas, proporcionando uma experiência de cultivo mais prática e menos dependente da supervisão constante dos utilizadores.

3.7. Promover Comunidades Verdes e Cidades Sustentáveis

- **Incentivar a criação de áreas verdes em zonas urbanas:** O **CultivUA** promove a transformação de espaços urbanos, como varandas e telhados, em áreas de cultivo, contribuindo para o aumento das áreas verdes e a melhoria da qualidade do ar nas cidades.

O **CultivUA** representa uma solução inovadora para os desafios da agricultura urbana, ao combinar tecnologia com a necessidade crescente de práticas sustentáveis e autossuficientes. Com esta abordagem prática e acessível, o projeto não só capacita os cidadãos a cuidarem das suas próprias plantas, como também promove o bem-estar pessoal e o desenvolvimento de cidades mais verdes e saudáveis. Ao integrar ferramentas tecnológicas como sensores IoT, uma loja online acessível e funcionalidades interativas, o **CultivUA** torna o processo de cultivo mais eficiente e adaptado às necessidades do utilizador moderno.

4. Planeamento

4.1. Cronograma e Recursos

No âmbito do projeto **CultivUA**, foi elaborado um cronograma. Este serviu como ferramenta essencial para planificar e organizar todas as diferentes etapas do desenvolvimento do projeto, garantindo que todas as tarefas fossem realizadas dentro dos prazos estabelecidos. De forma geral, o cronograma foi cumprido conforme o planeado, refletindo um esforço coletivo na gestão eficaz do tempo e dos recursos disponíveis.

A execução do projeto envolveu uma equipa composta por quatro pessoas, onde cada uma contribuiu ativamente em todas as fases de desenvolvimento. Para assegurar uma abordagem eficiente e equilibrada, as tarefas foram divididas de forma igualitária, tendo em conta as competências específicas e pontos fortes de cada membro da equipa. Este método de distribuição permitiu que cada tarefa fosse realizada com qualidade, aproveitando as melhores habilidades técnicas e criativas de cada um dos participantes.

Adicionalmente, foi adotada uma dinâmica de trabalho colaborativa, em que os quatro elementos da equipa trabalhavam em simultâneo em cada fase do projeto, promovendo um ambiente de apoio mútuo e de troca constante de ideias. Esta estratégia revelou-se essencial não apenas para cumprir os prazos definidos no cronograma, mas também para superar os desafios de forma mais ágil e eficaz.

4.1.1. Calendarização das tarefas prevista

Na fase de planeamento de um projeto, a elaboração de um diagrama de Gantt é fundamental para estruturar e gerir as tarefas durante a sua execução. Este processo implica definir a forma mais eficiente de distribuir as tarefas ao longo do tempo, levando em conta a duração de cada uma, as dependências entre elas e os prazos a serem cumpridos. Com base nesses elementos, foi criado um cronograma para o projeto, no qual estão indicadas as atividades planeadas e o tempo estimado para a sua realização.

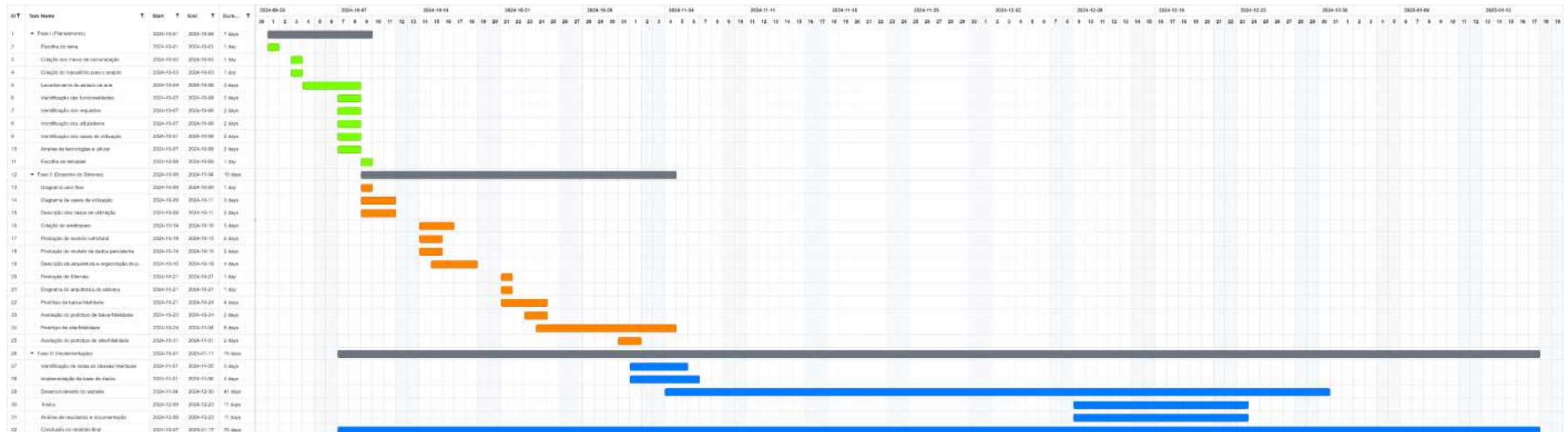


Figura 1 - Calendarização das tarefas prevista

4.1.2. Calendarização das tarefas executadas

Na fase de execução do projeto, o cronograma inicial sofreu alguns ajustes para refletir os imprevistos que foram surgindo ao longo do tempo. Alguns prazos foram estendidos e a alocação de recursos foi revista para atender às necessidades emergentes. As dependências entre as tarefas também foram reavaliadas, garantindo que as atividades seguissem uma sequência lógica e eficiente. Esses ajustes visaram otimizar o tempo e os esforços do grupo, de modo a cumprir os objetivos do projeto, apesar das mudanças que surgiram durante o desenvolvimento.

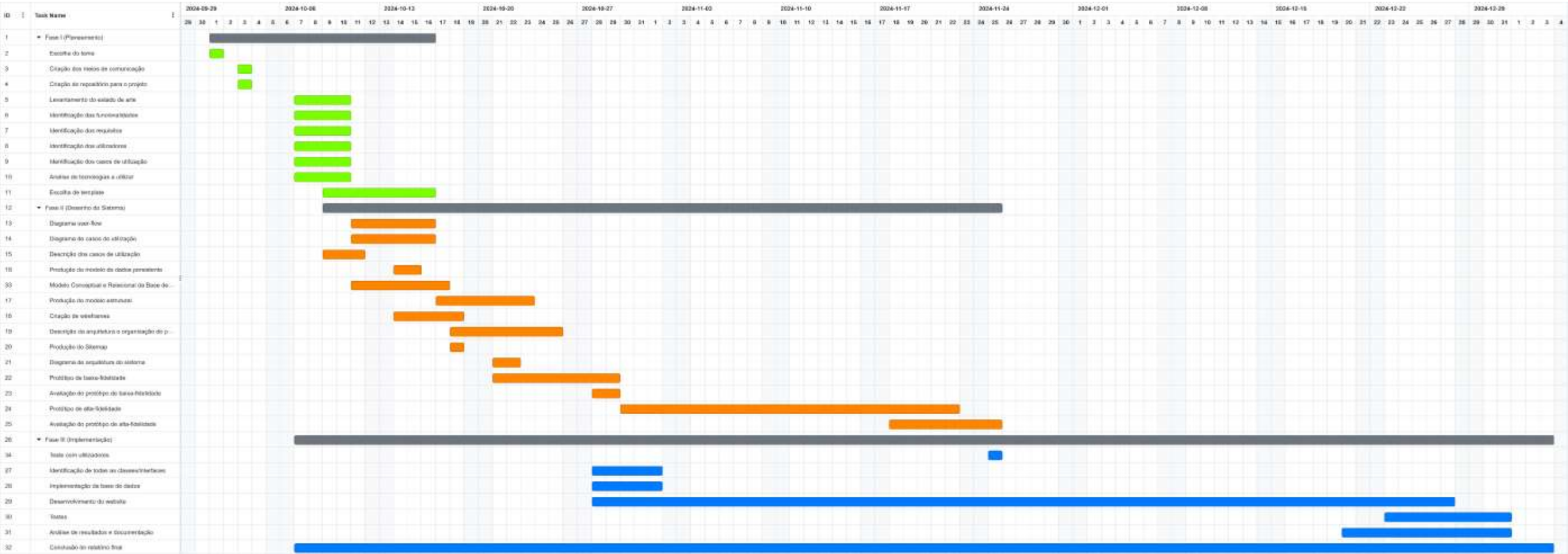


Figura 2 - Calendarização das tarefas executadas

4.1.3. Análise do Cronograma: Planeamento e Execução do Projeto

Em relação ao cronograma planeado e executado, constatou-se que, na fase I, algumas tarefas sofreram atrasos. Este atraso foi principalmente devido à necessidade de mais tempo para a definição inicial do projeto, que envolveu um maior esforço na análise e especificação dos requisitos.

No entanto, na fase II, decidimos ajustar a nossa abordagem e dedicar mais tempo ao planeamento e preparação das etapas subsequentes. Em particular, optámos por concentrar-nos no design do sistema, utilizando o **Figma** para criar protótipos. Decidimos também realizar o desenvolvimento do site em paralelo com o processo de design, o que permitiu uma integração mais fluída entre a interface de utilizador e a implementação funcional, garantindo que a base do projeto fosse sólida desde o início.

Na fase III, conseguimos retomar o cronograma planeado, com a exceção de alguns pequenos atrasos na parte final. No entanto, esta fase foi marcada por um avanço significativo, já que conseguimos iniciar o desenvolvimento mais cedo do que o inicialmente previsto. Esta abordagem permitiu-nos ter mais tempo para testar, refinar e ajustar as funcionalidades do sistema, assegurando uma entrega mais eficiente e com melhor qualidade.

4.2. Metodologia

Para o desenvolvimento do projeto, foi adotado o modelo de desenvolvimento de software em cascata. Este modelo caracteriza-se por um processo sequencial, em que cada fase é concluída antes de se iniciar a seguinte. As etapas principais incluem Análise de Requisitos, Planeamento, Implementação e Manutenção, formando uma estrutura em que os resultados de uma fase servem de base para a próxima.

Uma das grandes vantagens deste modelo reside na sua simplicidade e clareza, o que facilita a documentação e o controlo de qualidade. Cada etapa é cuidadosamente planeada e documentada, garantindo um fluxo de trabalho organizado e previsível. No entanto, o modelo em cascata apresenta uma limitação importante: a dificuldade de adaptação a mudanças nos

requisitos ou no design após a conclusão de fases anteriores. Essa rigidez pode ser um obstáculo em projetos onde as necessidades evoluem frequentemente.

Apesar desta potencial desvantagem, no contexto do **CultivUA**, a aplicação do modelo em cascata foi eficaz, pois o projeto teve requisitos bem definidos desde o início e manteve uma consistência ao longo do seu desenvolvimento. A abordagem metódica do modelo permitiu à equipa focar-se na qualidade em cada etapa, assegurando a construção de uma solução robusta e alinhada com os objetivos iniciais.

4.3. Análise de Tecnologias

No desenvolvimento do projeto **CultivUA**, foram utilizadas diversas tecnologias para garantir o funcionamento eficiente do sistema, desde a interface do utilizador até à integração de hardware. Abaixo, descrevem-se as principais tecnologias utilizadas em cada uma das áreas chave do projeto.

4.3.1. Frontend: Angular

Motivo da Escolha: O Angular é um *framework* robusto e amplamente utilizado para o desenvolvimento de interfaces dinâmicas. Suporta a criação de *Single Page Applications* (SPA), que resultam numa navegação fluída e numa melhor experiência para o utilizador.

Vantagens:

- **Estrutura modular:** Facilita a organização e manutenção do código, essencial para grandes aplicações.
- **Data *binding* bidirecional:** Garante uma sincronização eficiente entre dados e interface, ideal para a atualização em tempo real de informações dos sensores.
- **Ferramentas nativas:** Oferece soluções integradas para desenvolvimento de formulários, rotas e consumo de APIs.
- **Suporte robusto da comunidade:** A grande comunidade de desenvolvedores e o apoio contínuo da Google garantem estabilidade e inovação constante.
- **Suporte a PWAs:** Permite o desenvolvimento de *Progressive Web Apps*, ideal para uma plataforma acessível em diversos dispositivos.

Desvantagens:

- **Curva de aprendizagem:** A curva de aprendizagem do Angular pode ser mais acentuada, especialmente em comparação com bibliotecas como o React, que são mais simples para iniciantes.

- **Peso do *framework*:** Para aplicações mais simples, o Angular pode ser mais pesado em termos de desempenho, exigindo otimizações adicionais.
- **Configuração complexa:** A configuração do Angular CLI e a criação de projetos pode ser mais complexa em comparação com alternativas como o Vue.js.

Justificação: O Angular é a escolha ideal para este projeto, pois oferece uma estrutura sólida para desenvolver uma interface interativa e dinâmica. A capacidade de atualizar constantemente os dados dos sensores em tempo real e a robustez do *framework* são essenciais para o sucesso do sistema de monitorização de plantações.

4.3.2. Backend: Laravel

Motivo da escolha: O Laravel é um *framework* PHP que segue uma arquitetura MVC (*Model-View-Controller*), sendo amplamente reconhecido pela sua simplicidade, eficiência e robustez no desenvolvimento *backend*. O Laravel oferece ferramentas nativas que facilitam a criação de APIs RESTful, essenciais para a comunicação entre o *frontend* e os dispositivos de hardware, como sensores e sistemas de monitorização.

Vantagens:

- **Ferramentas nativas para autenticação e segurança:** Inclui sistemas integrados para autenticação, permissões de utilizadores e segurança, fundamentais para proteger a aplicação e os dados do utilizador.
- **ORM (Eloquent):** O Eloquent é um *Object-Relational Mapper* (ORM) que facilita a interação com a base de dados de forma simples e intuitiva, permitindo ao desenvolvedor focar na lógica de negócios sem se preocupar com detalhes complexos da manipulação de dados.
- **Suporte a APIs *RESTful*:** O Laravel facilita a criação de APIs RESTful, permitindo uma integração eficiente com os sensores e outros componentes do sistema, o que é essencial para o monitoramento de dados em tempo real.
- **Escalabilidade:** O Laravel oferece escalabilidade elevada, sendo capaz de lidar com o crescimento da aplicação, com fácil integração de serviços como *queues* (filas de tarefas) e *caching* (memória cache), que melhoram a performance em sistemas de grande escala.
- **Facilidade de integração:** O Laravel torna fácil integrar a aplicação com serviços de email, notificações e outras funcionalidades essenciais para o funcionamento do CultivUA.

Desvantagens:

- **Desempenho sob alta carga:** Embora o Laravel seja eficiente, pode ser mais lento do que soluções como o Node.js ao lidar com um grande volume de requisições simultâneas. No entanto, esse problema pode ser mitigado com técnicas como *caching* e a utilização de *queues*.
- **Necessidade de otimização para grandes projetos:** Em projetos de grande escala, o Laravel pode exigir otimizações manuais para garantir o desempenho ideal, principalmente em relação ao processamento de requisições intensivas.

Justificação: O Laravel foi escolhido por ser um *framework* seguro, altamente eficiente e fácil de integrar com bases de dados e dispositivos de hardware. A capacidade de gerar APIs *RESTful* robustas, bem como a segurança e a escalabilidade, fazem desta opção a escolha ideal para gerir a lógica *backend* do projeto, pois atende às necessidades de monitorização e integração com sensores em tempo real.

4.3.3. Base de Dados: MySQL

Motivo da escolha: O MySQL é um sistema de gestão de bases de dados relacionais (RDBMS) amplamente utilizado e reconhecido pela sua fiabilidade. É particularmente indicado para armazenar dados estruturados, como informações de utilizadores, leituras de sensores e histórico de cultivo.

Vantagens:

- **Ampla compatibilidade** com diversas linguagens de programação e *frameworks*, como o Laravel, o que facilita muito a integração entre as tecnologias.
- **Desempenho eficiente** para aplicações de pequeno e médio porte, o que garante um bom tempo de resposta para a maioria das operações.
- **Suporte a consultas SQL avançadas**, permitindo a criação de relatórios detalhados e consultas complexas, essenciais para o processamento de dados analíticos, como as leituras dos sensores.

Desvantagens:

- **Escalabilidade limitada** para grandes volumes de dados, podendo tornar-se um desafio à medida que o sistema cresce, especialmente em projetos de maior escala.
- **Desempenho reduzido** em operações de escrita intensiva, o que pode ser um problema em sistemas com elevada frequência de atualizações de dados.

Justificação: O MySQL foi escolhido devido à sua fiabilidade e compatibilidade com as tecnologias utilizadas no **CultivUA**, bem como pela sua capacidade de suportar operações em tempo real, como as leituras dos sensores. É uma solução robusta e eficaz para o armazenamento de dados estruturados essenciais para o funcionamento da aplicação.

Integração de Hardware: Arduino e Sensores

Motivo da escolha: O Arduino é uma plataforma *open-source* fácil de programar, altamente flexível e amplamente utilizada para integrar sensores de temperatura, humidade e outros sensores ambientais essenciais para monitorizar as condições da plantação.

Vantagens:

- **Flexibilidade na programação**, o que permite uma personalização total da forma como os dados dos sensores são recolhidos e transmitidos.
- **Grande variedade de sensores disponíveis**, o que possibilita a adaptação do sistema a diferentes tipos de monitorização, como temperatura, humidade, luz, etc.
- **Facilidade de integração com redes Wi-Fi** (através de módulos como **ESP8266** ou **ESP32**), o que permite o envio dos dados recolhidos para a plataforma web de forma eficiente.
- **Custo acessível e grande suporte da comunidade**, o que torna o desenvolvimento e a manutenção do sistema mais económicos e viáveis.

Desvantagens:

- **Para projetos de larga escala**, pode ser necessário recorrer a hardware mais robusto ou a microcontroladores mais potentes para garantir desempenho e fiabilidade a longo prazo.
- **Limitado para processamento intensivo** ou tarefas complexas, dado que o Arduino é principalmente adequado para recolha de dados e não para processamento intensivo.

Justificação: O Arduino foi escolhido por ser uma solução económica e eficaz para a monitorização de parâmetros ambientais em tempo real. A sua flexibilidade, combinada com a vasta gama de sensores disponíveis, torna-o uma escolha ideal para o **CultivUA**, que permite integrar facilmente a plataforma com os sensores necessários para o sistema de monitorização de plantações.

4.3.4 APIs Externas: Plant.id e Perenual

Motivo da escolha: Para enriquecer a plataforma **CultivUA**, é necessário integrar dados de fontes externas que fornecem informações precisas e atualizadas sobre as condições ambientais e cuidados com as plantas. As APIs escolhidas foram: Plant.id[3] e Perenual[4], que fornecem dados essenciais para a monitorização das condições, a identificação e orientações sobre o cultivo das plantas.

Plant.id

- **Descrição:** API de identificação de plantas que utiliza inteligência artificial para reconhecer espécies vegetais a partir de fotos.
- **Vantagens:** Permite aos utilizadores identificar plantas com facilidade e obter informações detalhadas sobre cuidados e requisitos específicos.
- **Justificação:** Fundamental para ajudar os utilizadores a identificar as suas plantas e fornecer recomendações personalizadas de cuidados.

Perenual

- **Descrição:** API especializada na criação de questionários sobre plantas, com o objetivo de ajudar a recomendar plantas baseadas nas preferências e estilo de vida do utilizador.
- **Vantagens:** Fornece uma forma personalizada de sugerir plantas, ideal para o quiz interativo do **CultivUA**.
- **Justificação:** Essencial para o projeto, pois permite recomendações de plantas adequadas ao perfil do utilizador, baseadas nas suas respostas ao quiz.

4.3.5. Template Spike Admin Angular

Motivo da escolha: A template Spike Admin[5] Angular (versão PRO) foi selecionada por proporcionar uma interface moderna e responsiva, desenhada para aplicações de administração, com fácil integração com o *framework* Angular. Esta *template* oferece uma estrutura versátil que se alinha com as necessidades do projeto, permitindo uma apresentação clara e organizada dos dados.

Vantagens:

- **Design Intuitivo e Responsivo:** O Spike Admin apresenta uma interface amigável e adaptável a vários dispositivos, o que proporciona uma experiência de utilização fluida.

- **Componentes Avançados:** Inclui uma ampla gama de componentes úteis para *dashboards*, como gráficos, tabelas interativas, formulários, e menus, o que facilita a implementação de funcionalidades administrativas.
- **Integração com Angular:** Construído com Angular, permite integração direta e eficiente com o resto do sistema, aproveitando as funcionalidades do *framework* para criar uma interface dinâmica e interativa.
- **Documentação Completa:** O *template* vem com uma documentação detalhada e código organizado, o que facilita o processo de personalização e adaptações necessárias ao projeto.

Desvantagens:

- **Complexidade em Funcionalidades Avançadas:** Algumas funcionalidades avançadas requerem ajustes adicionais, o que representa um esforço extra de desenvolvimento.

Justificação:

O Spike Admin Angular (versão PRO) foi escolhido pela combinação de recursos avançados e integração do Angular, o que permite termos uma base do painel administrativo funcional e esteticamente apelativa e eficiente. A variedade de componentes e a flexibilidade da *template* contribuem para um desenvolvimento mais ágil e para uma interface de fácil utilização, fundamental para o bom funcionamento da aplicação.

4.3.6. Stripe

Motivo da escolha: O Stripe[6] foi integrado na loja online do projeto para processar os pagamentos de forma segura e eficaz, garantindo a proteção dos dados do utilizador e facilitando a organização das transações financeiras.

Vantagens:

- **Fiabilidade e Simplicidade:** A plataforma é amplamente reconhecida pela sua fiabilidade e facilidade de integração.
- **Gestão Segura dos Dados Sensíveis:** Todo o processamento e armazenamento das informações sensíveis, como dados de pagamento, são geridos diretamente pelo Stripe.
- **Emissão Automatizada de Faturas:** O Stripe permite emitir faturas automaticamente, garantindo conformidade com as normas fiscais de cada país.

Desvantagens:

- **Taxas de Transação:** O Stripe cobra taxas por cada transação processada, o que pode ser uma desvantagem para negócios com um grande volume de pagamentos ou margens de lucro estreitas.
- **Complexidade nas Funcionalidades Avançadas:** Embora a integração básica seja simples, funcionalidades avançadas, como subscrições e pagamentos recorrentes, podem exigir configurações mais complexas e conhecimento técnico adicional.
- **Limitações de Países e Moedas:** O Stripe não está disponível em todos os países e pode ter limitações em termos de suporte a determinadas moedas, o que pode ser um problema para empresas com uma clientela global.

Justificação: O Stripe foi escolhido devido à sua capacidade de simplificar o processo de checkout e oferecer um sistema seguro para o utilizador. No momento do pagamento, as informações do cliente, como nome, morada e contacto, são geridas pelo Stripe, enquanto os detalhes dos produtos, como nome, quantidade, preço e total, são enviados diretamente do nosso site. A utilização desta plataforma proporciona um processo de compra transparente, rápido e em conformidade com as leis fiscais aplicáveis.

4.3.7. Bump.sh

Motivo da escolha: O Bump.sh[7] foi adotado para documentar as rotas da API Laravel Sanctum pela sua facilidade de documentar e partilhar, garantindo que toda a estrutura da API esteja clara e acessível a todos os membros do grupo.

Vantagens:

- **Documentação Interativa:** Oferece uma forma visual e interativa de explorar as rotas da API, simplificando o entendimento da sua estrutura e funcionamento.
- **Facilidade de Manutenção:** A documentação gerada permite compreender rapidamente as interações entre as rotas e a base de dados, facilitando futuras melhorias ou alterações.
- **Integração com a Equipa:** Ajuda novos membros a se familiarizarem com a API, acelerando o processo de *onboarding* e contribuindo para a eficiência do desenvolvimento.

Desvantagens:

- **Limitações na Personalização:** As ferramentas têm opções limitadas de personalização na documentação, o que torna mais difícil adaptar para uma estrutura mais adaptada às necessidades específicas do projeto.
- **Dependência de Conexão com o GitHub:** O Bump.sh depende de integração com o GitHub para funcionar corretamente, o que é um problema se o repositório estiver privado ou se houver dificuldades com a sincronização.
- **Custo para Funcionalidades Avançadas:** Embora o Bump.sh tenha uma versão gratuita, algumas funcionalidades mais avançadas e de personalização estão disponíveis apenas nas versões pagas, o que é um obstáculo para projetos com orçamentos limitados.

Justificação: O Bump.sh foi escolhido pela sua capacidade de criar uma documentação clara e interativa das rotas utilizadas pelo Laravel Sanctum. Essas rotas são responsáveis por realizar operações como GET, POST, PUT e DELETE na base de dados. A ferramenta documenta automaticamente essas interações, permitindo que se visualize e teste as rotas de forma prática. A documentação criada, que pode ser vista em **Anexo I**, é um recurso valioso para atualizações e integrações futuras, promovendo um desenvolvimento mais ágil e eficiente.

4.4. Divisão de Tarefas

A organização eficiente do trabalho foi fundamental para garantir o sucesso do projeto. Neste tópico, apresentamos a divisão de tarefas realizada entre os membros do grupo, onde é detalhado as responsabilidades específicas de cada um nas diferentes fases do desenvolvimento da plataforma.

A tabela apresentada na Erro! A origem da referência não foi encontrada. com a divisão de tarefas e as respetivas responsabilidades pode ser encontrada na página abaixo.

Fase I - Planeamento	Responsabilidade (%)			
Escolha do Tema	25	25	25	25
Levantamento do Estado de Arte	40	20	20	20
Identificação das Funcionalidades	25	25	25	25
Identificação dos Requisitos	25	25	25	25
Identificação dos Utilizadores	100	0	0	0
Identificação dos Casos de Utilização	25	25	25	25
Análise de Tecnologias a Utilizar	0	50	50	0
Escolha do Template	25	25	25	25
Fase II - Desenho do sistema				
Desenvolvimento Diagrama User Flow	70	10	10	10
Desenvolvimento Diagrama de Casos de Utilização	25	25	25	25
Desenvolvimento Descrição dos casos de Utilização	25	25	25	25
Desenvolvimento dos Modelos da Base de Dados	30	20	30	20
Desenvolvimento da Arquitetura da Informação	10	10	70	10
Produção do Sitemap	25	25	25	25
Protótipo de Baixa Fidelidade	25	25	25	25
Diagrama de Arquitetura do Sistema	0	100	0	0
Protótipo de Alta Fidelidade	20	40	20	20
Fase III - Implementação				
Desenvolvimento do Diagrama MVC	0	50	0	50
Implementação da Base de Dados	50	16(6)	16(6)	16(6)
Desenvolvimento do Website				
Landingpage	60	20	20	0
Autenticação	50	0	50	0
Identificar Plantas através de Imagens	100	0	0	0
Adicionar Planta ao Utilizador	0	0	100	0
Dashboard de Utilizador	30	70	0	0
Características, Gráficos, Guias, Rega, ... das Plantas	60	40	0	0
Dashboard de Admin	0	0	100	0
CRUD de Produtos	0	0	70	30
CRUD de Categorias	0	0	80	20
CRUD de Localizações	0	0	0	100
Gerir Tickets	0	0	100	0
Gerir Blog	80	20	0	0
Blog	80	20	0	0
Quiz	100	0	0	0
Leituras dos Sensores Através do Arduino	0	100	0	0
Contactos	0	0	50	50
Perfil de Utilizador	33(3)	0	33(3)	33(3)
Loja Online	0	75	25	0
Newsletter	90	10	0	0
Notificações	100	0	0	0
Documentação da API	0	100	0	0
Testes de Usabilidade	25	25	25	25
Testes de Software	0	0	100	0
Documentação e Escrita do Relatório	25	25	25	25
	Daniel Silva	Lucas Duarte	Gabriel Cravo	Miguel Pirré

Figura 3 - Tabela de divisão de tarefas

5. Levantamento do Estado de Arte

5.1. FarmBot

O FarmBot[8] é uma solução inovadora de agricultura automatizada *open-source* que permite aos utilizadores cultivar hortas de forma totalmente robotizada. Este sistema combina um equipamento físico com software de controlo remoto, no qual um braço robótico executa tarefas como plantação de sementes, rega de plantas e monitorização do solo.

Funcionalidades Principais

- **Automatização Completa:** O FarmBot controla de forma automática o processo de plantação, rega e monitorização das plantas, reduzindo ao mínimo a necessidade de intervenção do utilizador.
- **Sistema *Open-Source*:** Sendo um sistema *open-source*, o FarmBot permite que entusiastas e desenvolvedores personalizem e adaptem a plataforma conforme as suas necessidades específicas.
- **Planeamento de Cultivo:** A interface online do FarmBot possibilita o planeamento detalhado da horta, permitindo definir o layout do espaço e programar tarefas para cada planta.
- **Integração IoT:** Equipado com sensores, o FarmBot monitoriza fatores ambientais como a humidade do solo e a incidência de luz solar, otimizando o cuidado das plantas com base nas condições reais.

Pontos Fortes

- **Elevada Automação:** O FarmBot automatiza praticamente todo o processo de cultivo, ideal para quem deseja reduzir ao máximo o trabalho manual.
- **Interface Visual e Planeamento Detalhado:** A interface intuitiva permite um planeamento visual das plantações, facilitando o acompanhamento das operações.
- **Integração de Hardware e Software:** A plataforma oferece uma solução integrada, onde hardware e software operam em sincronia para controlar todos os aspetos do cultivo.

Limitações

- **Foco em Hortas Externas:** O FarmBot é projetado principalmente para hortas ao ar livre, o que pode limitar a sua aplicabilidade para quem deseja cultivar plantas em ambientes internos ou urbanos.
- **Custo Elevado:** O investimento inicial para aquisição e instalação do FarmBot pode ser um obstáculo para utilizadores com orçamentos mais limitados.
- **Curva de Aprendizagem:** Por ser uma solução avançada, o FarmBot apresenta uma curva de aprendizagem relativamente acentuada, exigindo algum conhecimento técnico para a sua instalação e utilização.

O que o CultivUA Pode Oferecer

- **Acessibilidade e Flexibilidade de Cultivo:** Ao contrário do FarmBot, que se concentra em hortas robotizadas, o **CultivUA** oferece uma abordagem mais acessível e flexível, ajustada a diferentes tipos de cultivo, como jardins internos e hortas verticais, facilitando a adoção da agricultura urbana.
- **Interface Intuitiva e Simplicidade de Uso:** Com uma interface orientada para o utilizador comum, o **CultivUA** é ideal para aqueles que procuram soluções de cultivo automatizadas, mas sem a complexidade de um sistema robotizado.
- **Diversificação de Plantas e Recomendações Personalizadas:** O **CultivUA** não se limita à automação de tarefas, oferecendo ainda recomendações específicas para diferentes condições (como luz, temperatura e humidade), permitindo uma personalização eficaz dos cuidados com cada planta.

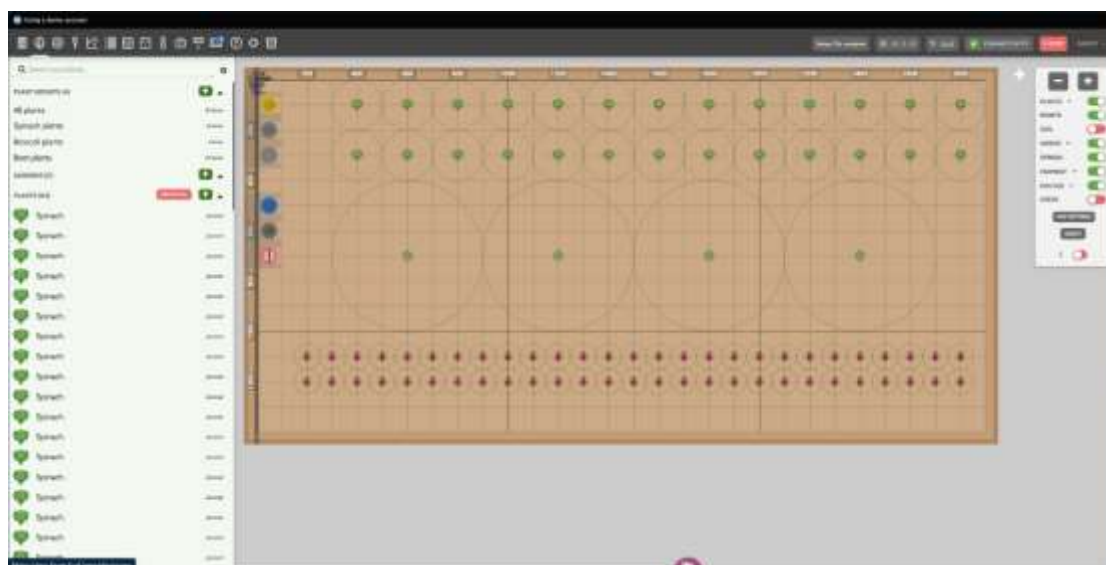


Figura 4 - FarmBot

5.2. Click & Grow

O Click & Grow[9] é um sistema de cultivo automatizado de pequena escala, ideal para o cultivo de plantas em espaços interiores. Desenvolvido com foco na simplicidade, utiliza cápsulas pré-carregadas com nutrientes e um sistema de irrigação automática, facilitando o cuidado de plantas para utilizadores em ambientes urbanos.

Funcionalidades Principais

- **Simplicidade de Uso:** Os utilizadores apenas precisam de colocar as cápsulas com sementes no sistema, que se encarrega das restantes etapas.
- **Manutenção Automática:** As necessidades de irrigação e iluminação das plantas são geridas automaticamente pelo sistema.
- **Aplicação Móvel:** A aplicação fornece informações sobre o estado das plantas e oferece dicas que ajudam a melhorar o cultivo.

Pontos Fortes

- **Facilidade de Utilização:** Ideal para quem deseja cultivar plantas sem investir tempo em cuidados diários.
- **Adequado para Espaços Urbanos Pequenos:** O design compacto é especialmente útil para apartamentos e outros espaços interiores limitados.
- **Acessibilidade para Iniciantes:** Não exige conhecimento prévio em jardinagem ou cultivo, facilitando o uso por qualquer pessoa.

Limitações

- **Sistema Fechado:** Os utilizadores ficam limitados ao uso exclusivo das cápsulas de sementes da marca, o que reduz as possibilidades de personalização e diversidade de plantas.
- **Dependência de Hardware Proprietário:** A utilização e expansão do sistema dependem do hardware da Click & Grow, o que limita a flexibilidade e as opções de adaptação.
- **Falta de Escalabilidade:** Menos indicado para utilizadores que procuram expandir para cultivos mais variados ou personalizáveis.

O que o CultivUA Pode Oferecer

- **Flexibilidade e Personalização:** Em contraste com o sistema fechado da Click & Grow, o **CultivUA** pode proporcionar uma experiência totalmente personalizável, permitindo que os utilizadores selecionem as espécies de plantas, sem a necessidade de cápsulas proprietárias.
- **Suporte para uma Maior Variedade de Plantas:** O **CultivUA** pode integrar sensores IoT para ajustar automaticamente as condições do ambiente de acordo com as necessidades específicas de uma vasta gama de plantas, oferecendo recomendações de cuidados para diferentes espécies, adaptadas ao contexto de cada utilizador.

5.3. Click & Grow – Loja Online

A Click & Grow[9] possui uma loja online bem estruturada e completa, onde comercializa os seus sistemas de cultivo, bem como acessórios e cápsulas de plantas. A interface da loja é moderna e intuitiva, proporcionando uma experiência de compra simples e integrada para os utilizadores.

Funcionalidades da Loja

- **Venda de Sistemas de Cultivo:** Oferece dispositivos de cultivo automatizado, como os vasos inteligentes Click & Grow.
- **Variedade de Cápsulas de Plantas:** A loja dispõe de uma ampla seleção de cápsulas com sementes pré-carregadas, desde ervas aromáticas a flores decorativas.
- **Subscrições:** Disponibiliza planos de subscrição, em que os utilizadores recebem regularmente cápsulas de plantas, incentivando a continuidade do uso do sistema.
- **Promoções Sazonais:** A loja realiza campanhas de descontos e promoções, promovendo a fidelização dos clientes.

Pontos Fortes

- **Experiência Integrada:** A loja online está desenhada para complementar o produto, garantindo que os utilizadores tenham acesso a tudo o que precisam para usar o sistema Click & Grow sem complicações.
- **Subscrições e Fidelização:** Os planos de subscrição ajudam a manter os clientes envolvidos e a promover o uso contínuo dos dispositivos.

O que o CultivUA Pode Oferecer na Loja Online

- **Ampla Variedade de Produtos:** Ao contrário do catálogo limitado da Click & Grow, o **CultivUA** pode diversificar a sua oferta, vendendo não apenas dispositivos de cultivo, mas também uma gama de produtos complementares, sementes, fertilizantes, vasos e ferramentas de cultivo. Essa variedade de opções permite ao **CultivUA** atender melhor as necessidades de utilizadores com diferentes objetivos e preferências de cultivo.

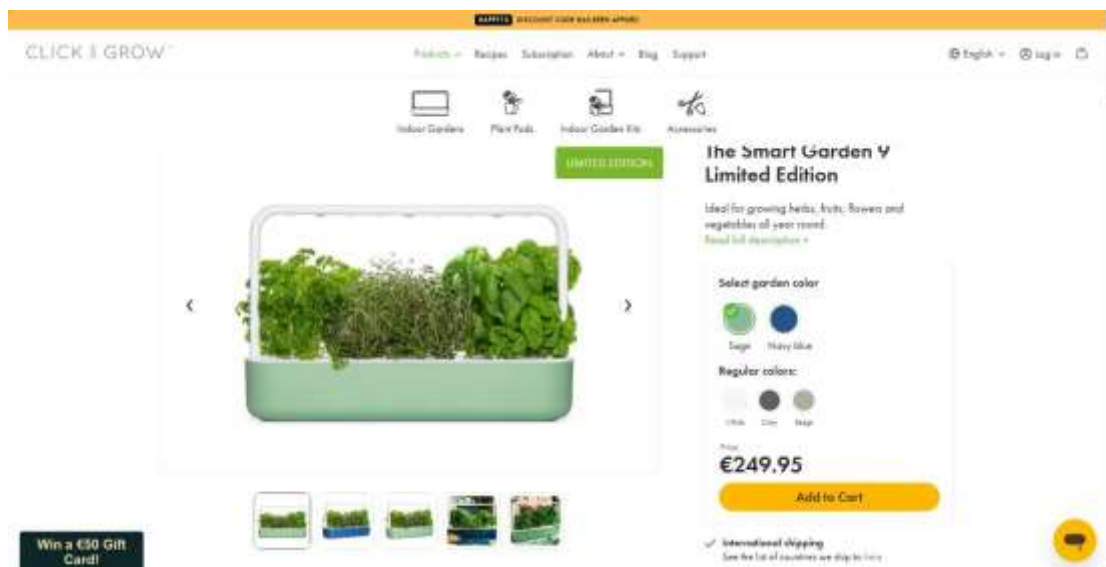


Figura 5 - Click & Grow

5.4. Gardena Smart System

O Gardena Smart System[10] é uma solução automatizada para gestão de jardins ao ar livre, com funcionalidades de irrigação inteligente, monitorização do solo e controlo remoto de dispositivos conectados. Destina-se principalmente a jardineiros e hortas urbanas em áreas exteriores, sendo ideal para quem tem espaços ao ar livre de maior dimensão.

Funcionalidades Principais

- **Controlo de Irrigação Inteligente:** Automatiza a rega com base nas condições do solo e nas previsões meteorológicas.
- **Sensores de Solo:** Mede a humidade e a temperatura do solo, ajustando a quantidade de água para maximizar a eficiência.
- **Aplicação Móvel:** Permite monitorizar e controlar remotamente os dispositivos e visualizar dados ambientais em tempo real.

Pontos Fortes

- **Gestão Abrangente de Jardins Externos:** Ideal para grandes jardins e quintais, oferecendo controlo sobre diversos dispositivos como aspersores e sensores.
- **Soluções Avançadas de Irrigação:** A integração com sensores e previsões meteorológicas otimiza o consumo de água, promovendo uma irrigação sustentável.
- **Integração com Dados Meteorológicos:** Ajusta automaticamente o sistema de irrigação conforme as previsões, poupando água em dias de chuva e otimizando a rega em dias secos.

Limitações

- **Dependência de Ambientes Externos:** A solução é pensada exclusivamente para espaços ao ar livre, não sendo prática para interiores ou pequenos espaços urbanos.
- **Custo Elevado dos Equipamentos:** O hardware especializado, como sensores de solo e dispositivos de irrigação, tem um preço elevado, limitando o acesso para utilizadores casuais.
- **Foco Limitado a Espaços Externos:** Com pouca adaptabilidade para o cultivo de plantas em ambientes internos, o sistema é restrito a áreas externas.

O que o CultivUA Pode Oferecer

- **Aplicação Adaptada a Ambientes Interiores e Exteriores:** Ao contrário do Gardena, que é focado em espaços ao ar livre, o **CultivUA** pode ser usado tanto em interiores quanto em exteriores, oferecendo uma solução versátil para vários tipos de ambiente.
- **Integração com Plantas de Pequeno Porte:** O **CultivUA** amplia as opções de cultivo, ao incluir plantas de interior, pequenos jardins verticais e outras formas de cultivo urbano que não exigem grandes áreas.
- **Monitorização mais acessível:** Recorrendo a soluções de monitorização com sensores IoT baseados em Arduino, o **CultivUA** pode proporcionar uma alternativa de baixo custo para monitorização de condições ambientais, tornando a tecnologia acessível a uma maior variedade de utilizadores.

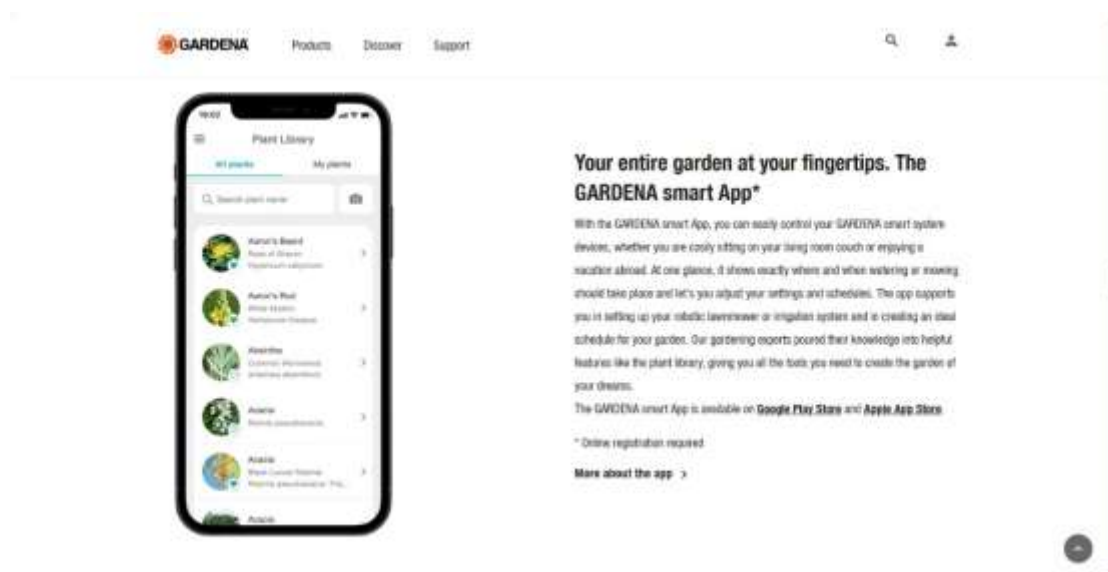


Figura 6 - Gardena Smart System

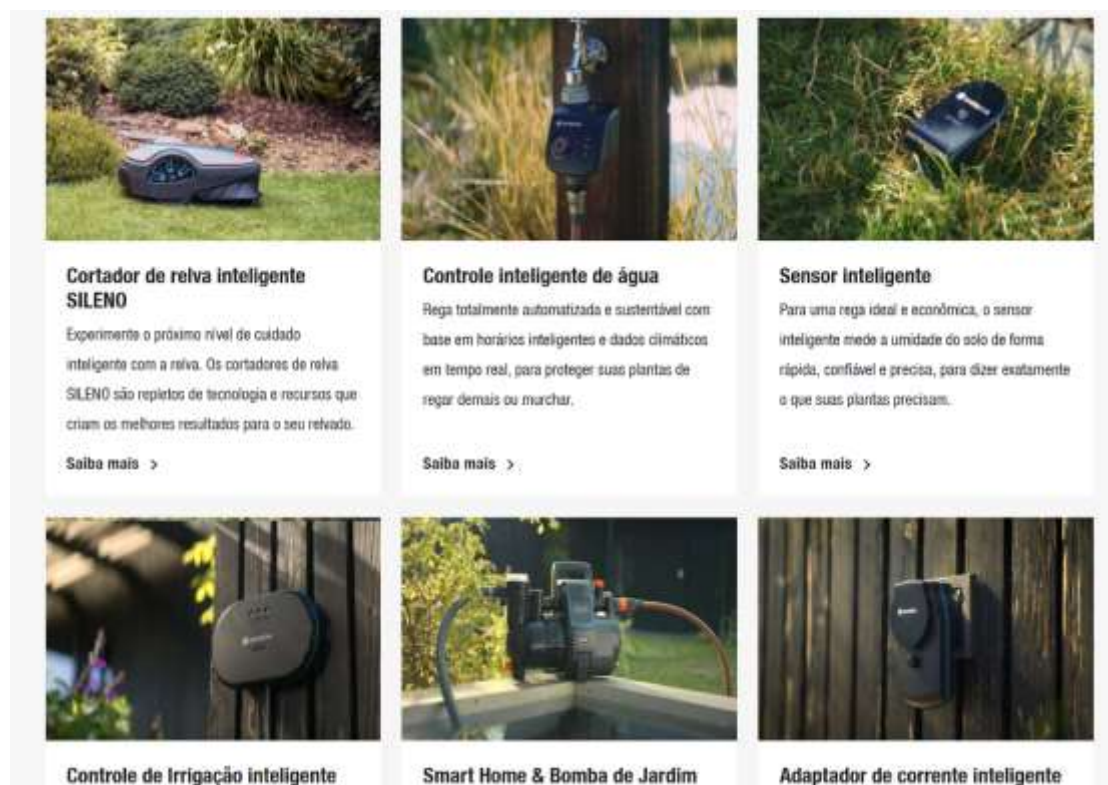


Figura 7 - Gardena Produtos

A análise dos sistemas existentes revela uma oportunidade para o **CultivUA** se destacar ao oferecer uma plataforma flexível, acessível e orientada para as necessidades de diferentes perfis de utilizador. Ao combinar monitorização IoT, uma loja online diversificada e uma interface intuitiva, o **CultivUA** não só preenche as lacunas das plataformas como FarmBot, Click & Grow e Gardena, mas também se adapta tanto a pequenos espaços interiores quanto a exteriores, ao promover a personalização e acessibilidade. Desta forma, o **CultivUA** posiciona-se como uma solução inovadora, económica e adaptada às necessidades da jardinagem urbana, sendo ideal para qualquer entusiasta de plantas.

6. Requisitos Funcionais

6.1. Loja Online

Navegação na Loja

- **RF.1:** O utilizador pode navegar pelos produtos da loja.
- **RF.2:** O utilizador pode visualizar os detalhes dos produtos (descrição, imagens, preços).
- **RF.3:** O utilizador pode filtrar e pesquisar produtos por tipo, preço, popularidade, etc.

Compra de Produtos

- **RF.4:** O utilizador pode adicionar produtos ao carrinho de compras.
- **RF.5:** O utilizador pode alterar a quantidade de itens no carrinho.
- **RF.6:** O utilizador pode visualizar o resumo da compra no carrinho.
- **RF.7:** O utilizador pode remover itens do carrinho.
- **RF.8:** O utilizador pode concluir a compra através de um sistema de checkout.
- **RF.9:** O utilizador pode inserir informações de pagamento e dados pessoais para envio.
- **RF.10:** O sistema confirma a compra e envia notificações por email com o resumo da compra.

Gestão de Stock

- **RF.11:** O administrador pode adicionar novos produtos à loja.
- **RF.12:** O administrador pode atualizar os detalhes dos produtos (preços, descrições, imagens).
- **RF.12:** O sistema atualiza automaticamente o stock de produtos após cada compra.
- **RF.13:** O sistema envia alertas ao administrador quando o stock de um produto estiver baixo.

6.2. Gestão de Perfis de Utilizador

Registo de Utilizadores

- **RF.14:** O utilizador pode criar uma conta na plataforma.
- **RF.15:** O utilizador pode registar-se utilizando email e palavra-passe.
- **RF.16:** O utilizador pode registar-se utilizando redes sociais (opcional).

Autenticação e Gestão de Conta

- **RF.17:** O utilizador pode fazer login na sua conta.
- **RF.18:** O utilizador pode recuperar a palavra-passe através de email.
- **RF.19:** O utilizador pode atualizar as suas informações pessoais (nome, morada, email).
- **RF.20:** O utilizador pode alterar a sua palavra-passe.
- **RF.21:** O utilizador pode visualizar o histórico de compras.

6.3. Registo e Gestão de Plantas

Registo de Plantas

- **RF.22:** O utilizador pode adicionar novas plantas ao seu perfil.
- **RF.23:** O utilizador pode inserir o nome, espécie e data de plantio de cada planta.
- **RF.24:** O sistema sugere cuidados com base na planta registada (via API externa).
- **RF.25:** O utilizador pode adicionar informações de acompanhamento, como rega, fertilização e poda.

Visualização do Histórico de Plantas

- **RF.26:** O utilizador pode visualizar o histórico de cuidados de cada planta (rega, poda, etc.).
- **RF.27:** O sistema guarda e exibe o histórico de crescimento da planta, com base nos dados recolhidos pelos sensores e nas atividades manuais.

6.4. Dashboard Interativa

Monitorização em Tempo Real

- **RF.28:** O utilizador pode monitorizar em tempo real (com intervalo de 1h) as condições das suas plantas (temperatura, humidade, luminosidade) através de gráficos na dashboard.
- **RF.29:** O sistema exibe as leituras dos sensores de temperatura, humidade do solo e luz.
- **RF.30:** O utilizador pode visualizar dados históricos e tendências de crescimento através de gráficos.

Notificações e Alertas

- **RF.31:** O utilizador recebe notificações automáticas via email ou na plataforma quando:
 - A planta precisa de água.
 - Os níveis de luz são insuficientes.
 - A humidade ou temperatura estão fora dos parâmetros ideais.
- **RF.32:** O utilizador pode configurar as preferências de alertas (frequência, canal de notificação).

Análise de Crescimento

- **RF.33:** O sistema gera relatórios de desempenho do crescimento das plantas.
- **RF.34:** O utilizador pode visualizar sugestões de otimização do ambiente de cultivo com base em dados recolhidos.
- **RF.35:** O sistema apresenta comparações de crescimento das plantas ao longo do tempo.

6.5. Sensores IoT e Arduino

Integração com Sensores

- **RF.36:** O sistema recolhe dados dos sensores de temperatura, humidade e luminosidade através do dispositivo Arduino.
- **RF.37:** O sistema integra-se com os sensores conectados via Wi-Fi.

Atualização Automática de Dados

- **RF.38:** O sistema atualiza automaticamente os dados recolhidos pelos sensores na dashboard do utilizador.
- **RF.39:** O administrador pode configurar os intervalos de recolha de dados dos sensores.

Monitorização Remota

- **RF.40:** utilizador pode monitorizar as condições das plantas remotamente através da web.
- **RF.41:** O sistema exibe dados em tempo real sobre as condições ambientais das plantas.

6.6. Integração com APIs Externas

Reconhecimento e Informação de Plantas

- **RF.42:** O sistema integra-se com APIs externas de reconhecimento de plantas, permitindo que o utilizador identifique espécies através de uma imagem.
- **RF.43:** O sistema consulta APIs externas para fornecer dados sobre os cuidados necessários para cada planta, incluindo:
 - Frequência de rega.
 - Necessidades de luz solar.
 - Dicas para otimizar o crescimento.

6.7. Gestão de Utilizadores Administradores

Gestão de Conteúdos

- **RF.44:** O administrador pode adicionar ou editar conteúdos informativos sobre cuidados com plantas na plataforma.
- **RF.45:** O administrador pode gerir as páginas de FAQ, política de privacidade e termos de serviço.

6.8. Relatórios e Estatísticas

Relatórios Personalizados

- **RF.46:** O utilizador pode gerar relatórios personalizados sobre o estado das suas plantas (baseados em dados históricos de crescimento e condições ambientais).
- **RF.47:** O sistema permite que o utilizador faça download dos relatórios em formatos PDF.

Estatísticas de Utilização

- **RF.48** O administrador pode visualizar estatísticas sobre o número de utilizadores, produtos mais vendidos, etc.
- **RF.49:** O sistema gera relatórios de desempenho da plataforma, incluindo tempo de resposta dos sensores e latência de dados.

7. Requisitos Não Funcionais

7.1. Performance

Tempo de Resposta

- **RNF.1:** O sistema deve garantir que os dados dos sensores sejam atualizados na *dashboard* em tempo real (latência máxima de 2 segundos).
- **RNF.2:** As páginas da loja online e da *dashboard* devem carregar em menos de 3 segundos para uma experiência de utilizador fluida.

Escalabilidade

- **RNF.3:** O sistema deve ser capaz de suportar um grande número de utilizadores simultâneos.
- **RNF.4:** O sistema deve ser dimensionado para integrar novos dispositivos IoT sem perda de desempenho.

7.2. Segurança

Proteção de Dados

- **RNF.5:** Todos os dados pessoais e de pagamento dos utilizadores devem ser encriptados.
- **RNF.6:** O sistema deve garantir a conformidade com o Regulamento Geral de Proteção de Dados (RGPD).
- **RNF.7:** Os dados dos sensores devem ser encriptados ao serem enviados para o servidor.

Autenticação e Acesso

- **RNF.8:** A plataforma deve implementar autenticação de dois fatores (2FA) para contas de utilizadores e administradores.
- **RNF.9:** Deve existir um sistema de gestão de permissões que restrinja o acesso a áreas administrativas apenas a utilizadores autorizados.

7.3. Usabilidade

Interface Intuitiva

- **RNF.10:** A plataforma deve ser fácil de usar, com uma interface clara e navegável tanto na loja online como na dashboard.
- **RNF.11:** A curva de aprendizagem para novos utilizadores deve ser mínima, com tutoriais ou dicas integradas.

Compatibilidade de Dispositivos

- **RNF.12:** O sistema deve ser responsivo e compatível com todos os tipos de dispositivos (desktop, tablet e smartphones).
- **RNF.13:** A plataforma deve funcionar nos principais browsers (Chrome, Firefox, Safari, Edge).

7.4. Manutenção

Atualizações Frequentes

- **RNF.14:** O sistema deve ser desenhado para permitir atualizações regulares de funcionalidades e correções de bugs sem impactar a experiência dos utilizadores.
- **RNF.15:** Atualizações de segurança devem ser aplicadas automaticamente sempre que necessário.

Monitorização e Logs

- **RNF.16:** O sistema deve incluir monitorização ativa para identificar e resolver problemas rapidamente (ex.: downtime, falhas de sensores).
- **RNF.17:** Devem ser mantidos logs de erros e atividades críticas do sistema para auditoria e análise de performance.

8. Requisitos tecnológicas e de alojamento

Para garantir o cumprimento dos objetivos do projeto **CultivUA** e a implementação eficiente das funcionalidades propostas, foi fundamental definir de forma rigorosa os requisitos tecnológicos e de alojamento. A escolha das tecnologias e ferramentas utilizadas teve como base critérios como compatibilidade, escalabilidade, facilidade de desenvolvimento e integração com os dispositivos de hardware.

8.1. Requisitos Tecnológicos

Foram integradas diversas tecnologias para oferecer uma plataforma funcional e robusta. Cada uma foi escolhida com base nas suas capacidades específicas de atender às necessidades do sistema:

- **Frontend - Angular:** Responsável por uma interface dinâmica e interativa.
- **Backend - Laravel:** *Framework* eficiente para gestão de APIs RESTful, autenticação segura e lógica de negócios, essencial para o processamento e integração de dados.
- **Base de Dados - MySQL:** Armazena de forma confiável os dados estruturados.
- **Hardware - Arduíno e Sensores:** Monitorizam parâmetros ambientais, integrando-se ao sistema de forma flexível e económica.
- **APIs Externas - Plant.id e Perennial:** Enriquecem a plataforma com funcionalidades de identificação de plantas e recomendações personalizadas de cuidados.

Mais detalhes sobre estas tecnologias, incluindo justificações, vantagens e desvantagens, encontram-se no tópico **4.3 Análise de Tecnologias**.

8.2. Desenvolvimento Local

Numa fase inicial de desenvolvimento, toda a aplicação foi configurada e executada localmente com a utilização do Docker[11], uma tecnologia de virtualização leve que permite criar e gerir ambientes isolados, conhecidos como *containers*. O Docker foi escolhido pela sua capacidade de garantir consistência no ambiente de desenvolvimento e pela facilidade de configuração e transporte das aplicações.

O *container* Docker incluía todos os componentes necessários para o funcionamento do sistema, incluindo uma base de dados MySQL[12] e o servidor *backend* em Laravel. Esta abordagem permitiu à equipa trabalhar de forma integrada e eficiente, garantindo que todos os elementos do sistema estavam devidamente sincronizados e operacionais no ambiente local.

Adicionalmente, foi utilizado o GitHub para a gestão do repositório de desenvolvimento, de forma a permitir o controlo de versões, colaboração entre os membros da equipa e a aplicação de boas práticas no desenvolvimento de software. A plataforma GitHub foi essencial para organizar o código, gerir alterações e assegurar que o progresso do projeto era devidamente documentado e acessível a todos.

8.3. Alojamento

Após a validação da aplicação em ambiente local, foi planeada a inserção da aplicação num ambiente de produção. Neste contexto, o professor orientador do projeto forneceu servidores da Universidade para alojar o projeto. No entanto, para garantir que a aplicação fosse acessível a todos os utilizadores, sem a necessidade de conexão à rede interna da Universidade, optou-se por implementar o alojamento com o auxílio de outras ferramentas.

A base de dados foi hospedada com a utilização do **Aiven**, que oferece estabilidade e escalabilidade para o armazenamento de dados. A aplicação Laravel, responsável pela Rest API, foi implantada no **Railway**, enquanto a interface Angular foi alojada no **Netlify**, proporcionando assim um desempenho otimizado e confiável.

Com esta abordagem, a plataforma encontra-se acessível online no endereço: <https://cultivua.netlify.app/>.

9. Identificação dos Utilizadores

9.1. Cidadãos Urbanos e Suburbanos Interessados em Agricultura

Descrição: Indivíduos que vivem em áreas urbanas ou suburbanas, com pouco ou nenhum espaço exterior, mas que desejam melhorar a sua qualidade de vida através do cultivo doméstico de alimentos frescos e plantas ornamentais. Estes utilizadores, muitas vezes sem experiência em jardinagem, procuram soluções práticas, acessíveis e tecnológicas para iniciar ou manter hortas urbanas.

Objetivos:

- Criar hortas urbanas em pequenos espaços como varandas, terraços ou janelas.
- Monitorizar facilmente as condições das plantas.
- Adquirir kits de cultivo prontos a usar.
- Aprender sobre jardinagem e agricultura urbana.
- Receber dicas e notificações automatizadas para o cuidado adequado das plantas.

Desafios:

- Falta de conhecimento prático sobre como começar a cultivar.
- Tempo limitado para dedicar ao cuidado das plantas.
- Espaço físico limitado para uma horta.

9.2. Entusiastas de Tecnologia e Agricultura

Descrição: Utilizadores que valorizam a integração de tecnologia e inovação no cultivo de plantas, com interesse em explorar sensores IoT, automação e relatórios avançados de dados. Este perfil inclui tanto iniciantes curiosos pela tecnologia quanto especialistas que desejam personalizar sistemas automatizados.

Objetivos:

- Utilizar tecnologia para monitorizar as plantas em tempo real.
- Explorar integrações com sistemas IoT e automação doméstica.
- Obter relatórios detalhados sobre o crescimento das plantas.

- Necessidade de um sistema altamente técnico e responsivo.

Desafios:

- Configurar e otimizar os sensores para uma monitorização eficiente.
- Necessidade de integração sem falhas entre sensores e a plataforma web.

9.3. Administradores da Plataforma

Descrição: Profissionais responsáveis pela gestão técnica e administrativa da plataforma, que incluem tarefas como atualização de conteúdos, gestão de utilizadores e produtos, e suporte técnico.

Objetivos:

- Gerir produtos, monitorizar o funcionamento da plataforma.
- Gerir utilizadores e fornecer suporte técnico.
- Atualizar informações e conteúdos relacionados com agricultura urbana.

Desafios:

- Garantir a disponibilidade e segurança da plataforma.
- Resolver problemas técnicos rapidamente.

9.4. Utilizadores com Necessidades Especiais

Descrição: Indivíduos com limitações físicas, sensoriais ou cognitivas que procuram uma experiência digital acessível e inclusiva.

Objetivos:

- Aceder à plataforma sem barreiras, independentemente do dispositivo utilizado.
- Obter informações e notificações através de interfaces adaptáveis.

Desafios:

- Interfaces não responsivas ou com falta de suporte para leitores de ecrã.
- Necessidade de navegação simplificada e intuitiva.

Funcionalidades Relevantes:

- Design responsivo e compatível com leitores de ecrã.
- Utilização de rótulos claros e acessíveis para navegação.
- Opções de personalização visual (contraste, tamanho de fonte).

9.5. Persona 1

Nome: Ana Oliveira

Idade: 34 anos

Profissão: Gestora de Marketing Digital

Localização: Lisboa, Portugal

Biografia:

Ana vive no centro de Lisboa, num apartamento com uma pequena varanda. Trabalha como gestora de marketing digital, passando a maior parte do tempo ao computador e em reuniões. Recentemente, interessou-se por alimentação saudável e sustentabilidade, a procurar envolver a sua família num estilo de vida mais verde. Com um horário de trabalho apertado, a Ana procura soluções práticas para cultivar as suas próprias ervas aromáticas e vegetais frescos, mas tem pouca experiência em jardinagem. O **CultivUA** chamou a sua atenção pela possibilidade de criar uma horta urbana com kits prontos, um quiz que a permite encontrar a planta ideal e monitorização através de uma plataforma intuitiva.

Metas e Motivações:

- Criar uma horta em casa para ter acesso a alimentos frescos e saudáveis.
- Ensinar o seu filho de 6 anos sobre a importância de cuidar da natureza e de onde vêm os alimentos.
- Manter a horta saudável com pouco esforço, dado o tempo limitado disponível.
- Receber notificações que a avisem quando é necessário regar ou ajustar a luz das plantas.

Frustrações e Desafios:

- Ana nunca fez jardinagem antes e sente-se insegura sobre como começar.
- Tem receio de falhar no cuidado das plantas devido à falta de conhecimento e tempo.
- A varanda do seu apartamento é pequena, o que limita as opções de cultivo.
- Procura soluções tecnológicas que simplifiquem a monitorização e manutenção da horta.

Necessidades e Expectativas do CultivUA:

- **Facilidade de Utilização:** Ana precisa de um sistema intuitivo que a guie desde a escolha do kit de cultivo até à monitorização das plantas.
- **Automatização:** Espera que os sensores e notificações ajudem a gerir a horta com o mínimo de intervenção.

Interligações com as funcionalidades do CultivUA:

- **RF.1 a RF.3:** A Ana precisa de navegar e visualizar os produtos na loja para encontrar kits de cultivo adequados à sua varanda. Isto facilita a seleção inicial de equipamentos e materiais para começar a horta.
- **RF.22 a RF.25:** O registo das plantas permite à Ana aprender sobre os cuidados específicos recomendados pelo sistema. Esta funcionalidade é essencial para ajudar alguém sem experiência a cuidar das plantas corretamente.
- **RF.31:** As notificações automáticas são cruciais para lembrar a Ana de ações como regar as plantas ou ajustar a luz, o que permite solucionar a falta de tempo e conhecimento, de forma a conseguir uma manutenção eficiente.
- **RF.42 e RF.43:** A consulta de informações detalhadas das plantas via APIs externas complementa a aprendizagem, com a oferta de práticas e guias de cuidado, ajudando a Ana a criar uma rotina eficaz.

9.6. Persona 2

Nome: João Martins

Idade: 42 anos

Profissão: Engenheiro de Software

Localização: Porto, Portugal

Biografia:

João trabalha remotamente como engenheiro de software e é um entusiasta de tecnologia e jardinagem. Tem uma casa com um pequeno jardim e está interessado em combinar as suas paixões por automação e cultivo de plantas. Com experiência em sensores IoT, João quer integrar soluções avançadas de monitorização e automação no seu jardim, utilizando sensores de luz, humidade e temperatura para maximizar o crescimento das plantas. O **CultivUA** interessa-lhe pela possibilidade da utilização de sensores para acompanhar os dados em tempo real através de uma *dashboard*.

Metas e Motivações:

- Otimizar o crescimento das suas plantas utilizando tecnologia IoT.
- Automatizar ao máximo o processo de monitorização e cuidado das plantas.
- Obter relatórios detalhados sobre as condições do solo e ambiente das plantas.
- Explorar novas funcionalidades tecnológicas que possam melhorar o cultivo.

Frustrações e Desafios:

- João quer garantir que os dados dos sensores são precisos e fáceis de monitorizar.
- O seu interesse por personalização leva-o a procurar funcionalidades avançadas que nem sempre encontra em plataformas simples.
- Ele deseja uma solução que permita expandir a sua rede de sensores sem complicações técnicas.

Necessidades e Expectativas do CultivUA:

- **Tecnologia Avançada:** João espera que a plataforma seja altamente técnica e integrada com soluções IoT.
- **Personalização:** Quer uma solução que lhe permita adaptar os sensores e otimizar os dados recolhidos para melhorar a eficiência do seu jardim.

Interligações com as funcionalidades do CultivUA:

- **RF.28 a RF.30:** O João utiliza a monitorização em tempo real para acompanhar o crescimento e condições das plantas. Isto é essencial para obter dados imediatos e ajustar as condições de forma otimizada.
- **RF.36 a RF.41:** A integração e atualização automática dos sensores IoT são fundamentais para o João configurar o seu próprio sistema técnico e responsivo, o que lhe permite implementar as automações que procura, eliminando tarefas manuais.
- **RF.33 a RF.35:** A visualização de relatórios e gráficos detalhados sobre as condições e crescimento complementa a análise técnica e dá-lhe o controlo total sobre a performance das plantas.
- **RF.42:** A identificação de plantas por imagem permite-lhe explorar novas espécies e integrar mais sensores para estas. Assim, é encorajada a expansão e personalização do seu sistema de monitorização.

10. Casos de utilização.

Os casos de utilização são uma ferramenta essencial para descrever as interações possíveis entre os utilizadores e o sistema, que detalham os cenários em que o **CultivUA** será utilizado e as funcionalidades disponibilizadas. Este tópico tem como objetivo apresentar de forma clara e estruturada as principais operações que podem ser realizadas na plataforma, desde a perspetiva dos diferentes tipos de utilizadores.

Ao identificar os casos de utilização, é possível compreender como o sistema responde às necessidades dos utilizadores, antecipar eventuais problemas, e garantir que todas as funcionalidades são implementadas de forma a proporcionar uma experiência eficiente, intuitiva e prática. Estes casos baseiam-se nas funcionalidades do sistema, que incluem a monitorização de hortas urbanas através de sensores, a consulta de informações sobre plantas, e a gestão das configurações da aplicação.

Além disso, os casos de utilização ajudam a alinhar as expectativas da equipa de desenvolvimento com os objetivos do projeto, servindo assim como uma referência importante tanto para a implementação do sistema como para os testes e validação das funcionalidades.

Nos subcapítulos seguintes, serão apresentados os principais casos de utilização identificados para o projeto. Caso seja necessário, o restante dos casos de utilização completos pode ser consultado no **Anexo C**, enquanto o diagrama dos mesmos está disponível no Anexo B, oferecendo uma visão detalhada de cada funcionalidade e do seu impacto no sistema.

10.1. Loja Online

Adicionar Produto ao Carrinho

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador pode selecionar um produto e adicioná-lo ao carrinho de compras. A quantidade pode ser ajustada conforme necessário.
- **Pré-condições:** O utilizador deve estar autenticado.

- **Fluxo Principal:**
 1. O utilizador navega pela loja online e seleciona um produto.
 2. O utilizador escolhe a quantidade desejada.
 3. O utilizador clica no botão "Adicionar ao Carrinho".
 4. O sistema adiciona o produto ao carrinho e atualiza a quantidade.
- **Fluxo Alternativo:**
 - O produto não está disponível em stock. O sistema exibe uma mensagem de erro.
- **Pós-condições:** O produto é adicionado ao carrinho do utilizador.

Checkout

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador conclui a compra inserindo as suas informações pessoais e de pagamento.
- **Pré-condições:** O utilizador tem produtos no carrinho.
- **Fluxo Principal:**
 1. O utilizador clica em "Finalizar Compra".
 2. O utilizador insere dados de envio e pagamento.
 3. O sistema processa o pagamento e exibe uma confirmação.
 4. O utilizador recebe um email de confirmação.
- **Fluxo Alternativo:**
 - O pagamento falha. O sistema exibe uma mensagem de erro e pede para tentar novamente.
- **Pós-condições:** A compra é finalizada e os produtos serão enviados.

Pagamento de Produtos

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador paga pelos produtos ao utilizar métodos como cartão de crédito ou transferência bancária.
- **Pré-condições:** O utilizador tem produtos no carrinho e inseriu informações de pagamento.
- **Fluxo Principal:**
 1. O utilizador seleciona o método de pagamento preferido.
 2. O utilizador insere os detalhes do pagamento.

3. O sistema processa o pagamento.
 4. O utilizador recebe uma notificação de pagamento bem-sucedido.
- **Fluxo Alternativo:**
 - O pagamento é recusado. O sistema solicita a correção dos dados de pagamento.
 - **Pós-condições:** O pagamento é concluído.

10.2. Gestão de Utilizadores

Registar-se

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador pode criar uma conta ao fornecer um email e uma palavra-passe.
- **Pré-condições:** O utilizador não tem uma conta existente.
- **Fluxo Principal:**
 1. O utilizador clica em "Registar-se".
 2. O utilizador preenche o formulário de registo com email e palavra-passe.
 3. O sistema cria a conta e envia um email de confirmação.
- **Fluxo Alternativo:**
 - O email já está em utilização. O sistema exibe uma mensagem de erro.
- **Pós-condições:** O utilizador tem uma conta criada.

Autenticar

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador entra na plataforma com email e palavra-passe.
- **Pré-condições:** O utilizador tem uma conta.
- **Fluxo Principal:**
 1. O utilizador insere as suas credenciais.
 2. O sistema valida as credenciais e autentica o utilizador.

- **Fluxo Alternativo:**
 - As credenciais são inválidas. O sistema exibe uma mensagem de erro.
- **Pós-condições:** O utilizador está autenticado.

10.3. Monitorização e Gestão de Plantas

Monitorizar Plantas

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador pode monitorizar as condições das suas plantas (temperatura, humidade, luminosidade) através de gráficos na *dashboard*, com dados fornecidos pelos sensores conectados.
- **Pré-condições:** O utilizador tem plantas e sensores registados.
- **Fluxo Principal:**
 1. O utilizador acede à secção de monitorização de plantas.
 2. O sistema apresenta gráficos e dados atualizados dos sensores conectados.
- **Pós-condições:** O utilizador visualiza os dados das plantas em tempo real.

Receber Notificações e Alertas

- **Ator Principal:** Utilizador
- **Descrição:** O sistema envia notificações ao utilizador quando as plantas precisam de água, luz ou se os parâmetros ambientais estão fora dos níveis ideais.
- **Pré-condições:** O utilizador tem plantas registadas e sensores instalados.
- **Fluxo Principal:**
 1. O sistema monitoriza os sensores conectados.
 2. O sistema identifica se há condições fora dos parâmetros.
 3. O sistema envia uma notificação ao utilizador.
- **Fluxo Alternativo:**
 - O utilizador não ativou notificações. O sistema não envia alertas.
- **Pós-condições:** O utilizador é notificado sobre o estado das plantas.

Visualização do Histórico de Plantas

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador pode visualizar o histórico completo das condições de cada planta com base em dados recolhidos pelos sensores e registos manuais.
- **Pré-condições:** O utilizador tem plantas registadas e sensores instalados.
- **Fluxo Principal:**
 1. O utilizador acede ao histórico de uma planta específica.
 2. O sistema apresenta gráficos e dados detalhados sobre a condição da planta.
- **Pós-condições:** O utilizador visualiza o histórico da condição da planta.

10.4. Sensores

Enviar Notificações de Alerta para o Sistema

- **Ator:** Sensores
- **Descrição:** O sistema envia notificações de alerta para o utilizador sempre que os sensores detetarem condições adversas nas hortas (como níveis baixos de humidade, temperatura inadequada ou falta de luz).
- **Pré-condições:** Os sensores IoT estão instalados e monitorizando as condições da horta em tempo real.
- **Fluxo Principal:**
 1. Os sensores captam dados em tempo real (ex.: humidade do solo, temperatura do ar).
 2. O sistema identifica uma condição crítica (ex.: humidade abaixo do nível ideal).
 3. O sistema envia uma notificação de alerta ao utilizador através da plataforma web ou aplicação móvel.
- **Pós-condições:** O utilizador é notificado e pode agir para corrigir o problema, como ajustar a irrigação ou proteger as plantas.

Monitorizar Condições da Planta

- **Ator:** Sensores
- **Descrição:** O utilizador pode monitorizar as condições de crescimento das plantas, como humidade do solo, temperatura ambiente e luz disponível, através de uma interface na plataforma web.
- **Pré-condições:** Os sensores IoT estão em operação e a plataforma está disponível para o utilizador.
- **Fluxo Principal:**
 1. Os sensores IoT recolhem dados das condições ambientais e do solo ao redor das plantas.
 2. O sistema processa os dados e atualiza a *dashboard* na plataforma web em tempo real.
 3. O utilizador acede à plataforma para visualizar o estado atual das condições de cultivo.
- **Pós-condições:** O utilizador tem acesso às informações sobre as condições da horta, permitindo que faça ajustes para otimizar o crescimento das plantas.

11. Modelos de base de dados

Os modelos de base de dados desempenham um papel essencial no desenvolvimento de sistemas como o **CultivUA**. Estes definem a estrutura lógica e relacional que permite o armazenamento e a recuperação eficiente de informações. Neste projeto, foi adotado o sistema de gestão de bases de dados relacional **MySQL**, amplamente reconhecido pela sua robustez e compatibilidade com *frameworks*, como o Laravel.

O **CultivUA** beneficia de uma modelagem bem estruturada para assegurar a integridade dos dados e a escalabilidade da aplicação, incorporando princípios de normalização e boas práticas de design de base de dados. Esta secção detalha os principais conceitos e decisões envolvidos na conceção dos modelos de dados.

11.1. Modelo Conceptual

O modelo conceptual traduz as necessidades de armazenamento e relacionamentos do domínio de agricultura urbana para um formato de alto nível, independente de implementação técnica. Este modelo descreve as entidades, os atributos e as relações, destacando:

- **Entidades principais:** Utilizadores, Plantas, Produtos, Pedidos, Kits (de sensores), Notificações, entre outros.
- **Atributos relevantes:** Cada entidade possui atributos que representam as suas propriedades essenciais, como a temperatura, humidade, luz e rega ideais, na entidade *Plantas*.
- **Relacionamentos:** As relações entre entidades são expressas de forma clara, como por exemplo:
 - 1:N entre *users* e *users_plants*: Um utilizador pode registar várias plantas através da tabela *users_plants*.
 - 1:N entre *plants* e *users_plants*: Uma planta pode ser associada a várias entradas em *users_plants* para diferentes utilizadores.
 - 1:N entre *users_plants* e *watering_history*: O histórico de rega é associado a plantas específicas dos utilizadores.
 - 1:1 entre *users_plants* e *kits*: Uma planta registada por um utilizador só pode estar associada a um *kit*, e um *kit* só pode ser utilizado por uma planta.

Este modelo é visualizado através do diagrama ER (entidade-relação) seguinte:

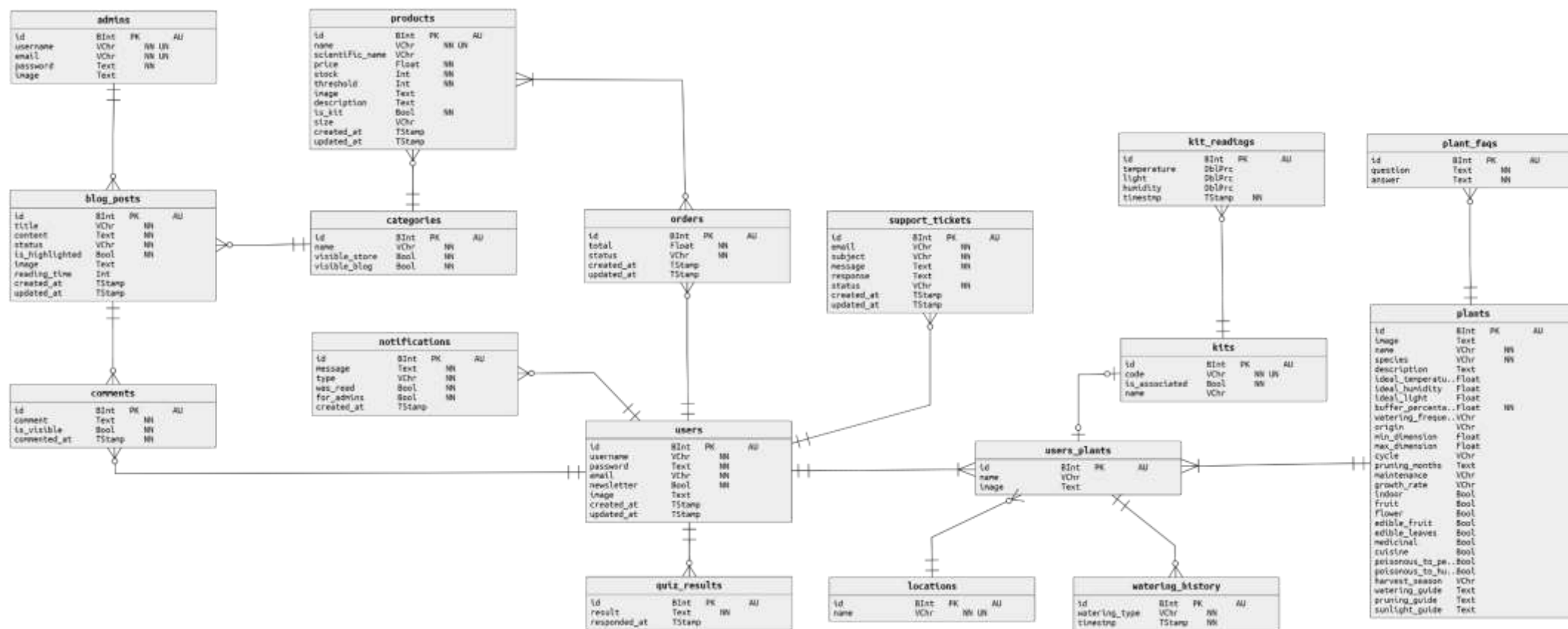


Figura 8 - Modelo Conceptual da Base de Dados

11.2. Modelo Relacional

O modelo relacional do **CultivUA** implementa o modelo conceptual em tabelas normalizadas, assegurando eficiência e evitando redundâncias. Com chaves primárias e estrangeiras que asseguram a integridade referencial, relações entre entidades identificadas, de forma a fornecer melhor legibilidade e restrições *UNIQUE* e *NOT NULL*, que garantem consistência nos dados.

Foram adotadas boas práticas de normalização, atingindo a Terceira Forma Normal (3NF), isto é:

- Cada atributo é dependente exclusivamente da chave primária.
- Não há dependências transitivas.
- Esta abordagem minimiza redundâncias, aumenta a integridade dos dados e melhora o desempenho nas consultas.

De seguida, será apresentado o diagrama entidade-relação que representa o modelo relacional da base de dados. O *script SQL* de criação das tabelas será disponibilizado no **Anexo**.

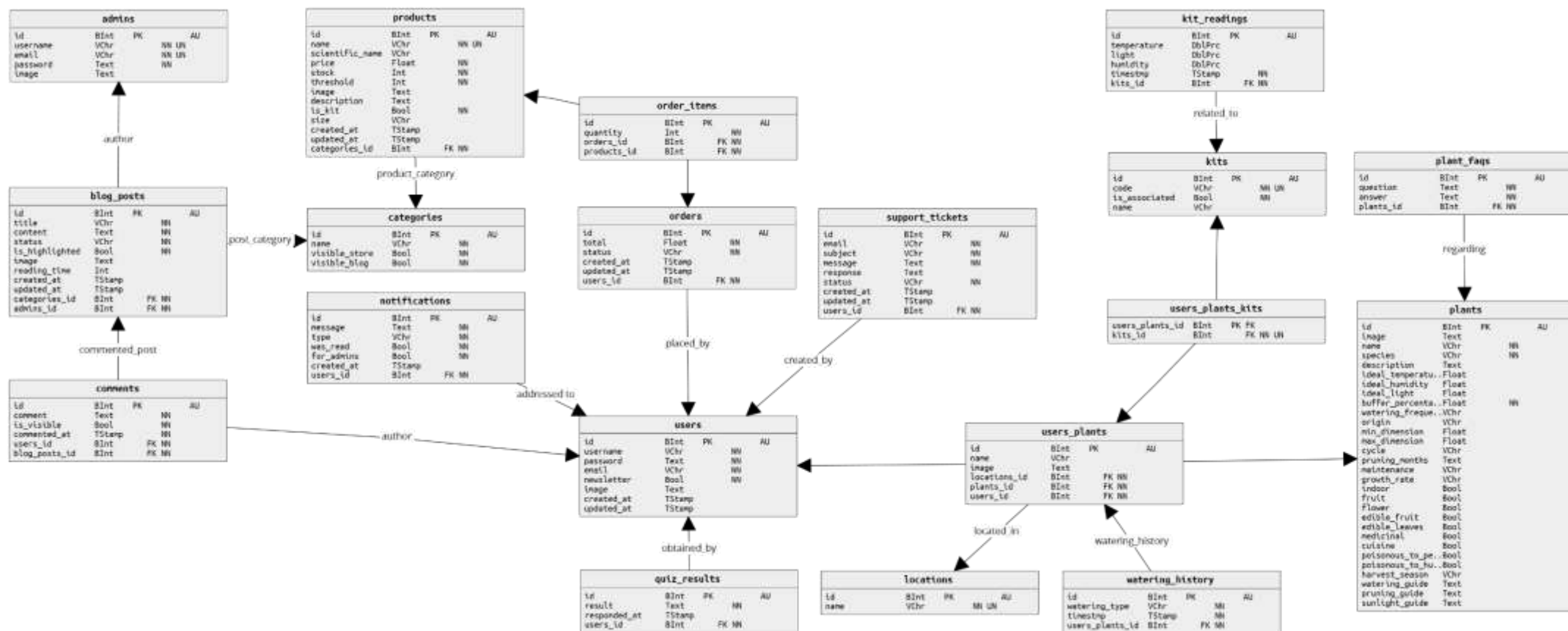


Figura 9 - Modelo Relacional da Base de Dados

11.3. Otimização da Base de Dados com Índices

A implementação de índices na base de dados é uma das principais estratégias para melhorar o desempenho das consultas e garantir a eficiência dos *triggers* que dependem de leituras frequentes sobre as tabelas.

Um índice é uma estrutura de dados que permite aceder de forma mais rápida aos registos das tabelas, reduzindo a carga computacional em operações de pesquisa, filtragem e *joins*.

Índices Implementados

Tabela *users*:

- **Índice em *id***: Facilita pesquisas primárias e operações de *join* em tabelas associadas, como *notifications* e *orders*.
- **Índice em *email***: Acelera autenticações e validações de e-mails únicos, garantindo integridade nas operações de registo e login.

Tabela *plants_kits*:

- **Índices em *user_plant_id* e *kits_id***: Otimizam os *joins* frequentes entre plantas associadas a utilizadores e kits, melhora o desempenho na identificação de associações *muitos-para-muitos*.

Tabela *kit_readings*:

- **Índice em *timestmp***: Aumenta a eficiência de consultas cronológicas, como análise de leituras históricas e geração de relatórios e gráficos baseados em intervalos de tempo.

Tabela *products*:

- **Índice em *is_kit***: Permite distinguir rapidamente produtos que são kits de cultivo, que é essencial para o *trigger* responsável por criar novos kits com base nos pedidos da tabela *order_items*.

Tabela *kits*:

- **Índice em *código***: Acelera as verificações de associação e disponibilidade de kits, crucial para garantir a criação de códigos únicos e evitar duplicações no *trigger* que gera novos kits.

Os índices descritos foram essencialmente projetados para otimizar os *triggers* da base de dados, que desempenham um papel muito importante no funcionamento automatizado do sistema.

Por exemplo:

- O índice em `is_kit` melhora a verificação no *trigger* que cria kits automaticamente ao processar pedidos.
- O índice em *timestamp* acelera consultas em *triggers* como o `detect_watering`, que avalia variações significativas nas leituras de humidade.
- Índices em tabelas de associação, como `plants_kits`, garantem eficiência em operações complexas que envolvem *joins*.

A utilização destes índices reduziu significativamente a latência dos *triggers*.

11.4. Automatização do Sistema com Triggers

Os *triggers* são elementos fundamentais para a automatização de processos e para a manutenção da integridade e consistência dos dados numa base de dados. No **CultivUA**, os *triggers* desempenham um papel crucial, onde automatizam notificações, verificações e atualizações, de maneira a reduzir o esforço manual e a garantir um comportamento reativo e eficiente.

Triggers Implementados

1. Trigger `after_kit_readings_insert`: Notificações de Cuidados com Plantas

- **Descrição:** Este *trigger* é acionado após a inserção de uma nova leitura na tabela `kit_readings`. Este verifica se as condições ambientais de uma planta (temperatura, humidade e luz) estão fora dos limites aceitáveis, considerando um buffer calculado a partir das condições ideais.
- **Função:** Se uma condição estiver fora dos limites, uma notificação é gerada e armazenada na tabela *notifications*, alertando o utilizador para a necessidade de intervenção.
- **Vantagem:** Automatiza a monitorização de condições ambientais, permitindo respostas rápidas e otimizando o cuidado com as plantas.

2. Trigger `after_blog_post_insert`: Notificações de Novos Posts no Blog

- **Descrição:** Este *trigger* é ativado sempre que um novo *post* é criado na tabela `blog_posts`. Ele notifica automaticamente todos os utilizadores subscritos à newsletter sobre a nova publicação.

- **Função:** Insere notificações na tabela *notifications* para utilizadores com a opção de *newsletter* ativada.
- **Vantagem:** Melhora a comunicação com os utilizadores e incentiva o envolvimento com o conteúdo do blog.

3. **Trigger after_stock_update: Notificações de Stock Baixo**

- **Descrição:** Acionado após uma atualização na tabela *stock_products*, este *trigger* verifica se o stock de um produto caiu abaixo de um limite pré-definido (*threshold*).
- **Função:** Gera notificações para administradores, alertando sobre a necessidade de reabastecimento do produto.
- **Vantagem:** Garante a manutenção de níveis adequados de stock, evitando situações de rutura.

4. **Trigger after_ticket_insert: Notificações de Novos Tickets de Suporte**

- **Descrição:** Este *trigger* é acionado após a criação de um novo ticket na tabela *support_tickets*. Este notifica os administradores sobre a nova solicitação de suporte.
- **Função:** Insere notificações na tabela *notifications*, garantindo que as solicitações sejam tratadas rapidamente.
- **Vantagem:** Melhora o tempo de resposta do suporte e a experiência do utilizador.

5. **Trigger after_order_items_insert: Gestão de Stock em Pedidos**

- **Descrição:** Este *trigger* verifica se o stock de um produto é suficiente para atender a um pedido na tabela *order_items*. Caso contrário, lança um erro.
- **Função:** Atualiza o stock na tabela *products*, subtraindo a quantidade encomendada.
- **Vantagem:** Evita pedidos inválidos e mantém a consistência do stock em tempo real.

6. **Trigger detect_watering: Registo de Eventos de Rega Automática**

- **Descrição:** Após uma nova leitura na tabela *kit_readings*, este *trigger* verifica se houve um aumento significativo na humidade, indicando um evento de rega.
- **Função:** Regista a rega na tabela *watering_history* como um evento automático.

- **Vantagem:** Automatiza o acompanhamento de regas, permitindo uma análise mais detalhada do histórico de cuidados.

7. **Trigger after_order_insert: Geração Automática de Kits**

- **Descrição:** Acionado quando um item de pedido referente a um kit é inserido na tabela `order_items`. Este *trigger* verifica se o item é um kit e, caso seja, gera automaticamente novos kits com códigos únicos.
- **Função:** Gera códigos através de um algoritmo que evita duplicação e cria novos registos na tabela `kits`.
- **Vantagem:** Automatiza a gestão de kits, reduzindo o trabalho manual e garantindo unicidade nos códigos.

Benefícios Gerais dos Triggers

- **Automatização:** Reduz operações manuais, otimizando fluxos de trabalho.
- **Consistência:** Garante que as regras de negócio sejam aplicadas uniformemente.
- **Performance:** Combinados com índices, os *triggers* garantem baixa latência mesmo sob cargas intensas.
- **Monitorização Proativa:** Identifica e reage a eventos críticos em tempo real.

Os *triggers* implementados no **CultivUA** não só aumentam a eficiência operacional da plataforma, mas também melhoram a experiência do utilizador, promovendo um sistema confiável e responsivo.

12. Arquiteturas de Informação

A arquitetura de informação é o pilar que sustenta a organização e estruturação dos conteúdos, funcionalidades e sistemas de navegação de um website [13]. No contexto do **CultivUA**, esta estrutura foi projetada para proporcionar uma experiência intuitiva, eficiente e acessível, alinhada com os objetivos do projeto e as necessidades dos seus utilizadores

12.1. Objetivos da Arquitetura de Informação

A arquitetura de informação do **CultivUA** foi desenhada para:

1. **Facilitar a navegação:** Organizar o conteúdo em categorias lógicas que maximizem a eficiência de acesso.
2. **Garantir usabilidade:** Permitir que utilizadores com diferentes níveis de experiência tecnológica acessem facilmente às funcionalidades.
3. **Promover acessibilidade:** Assegurar que todo o sistema seja responsivo e compatível com diferentes dispositivos, atendendo a utilizadores com necessidades especiais.
4. **Sustentar escalabilidade:** Suportar a integração de futuras funcionalidades sem comprometer a experiência do utilizador.

12.2. Estruturas de Informação

A organização hierárquica do **CultivUA** é evidente nas suas principais secções: ***Dashboard do Administrador***, ***Dashboard do Utilizador***, ***Loja Online***, ***Quiz***, ***Blog***, e ***Página Inicial***. Abaixo, cada secção é detalhada.

Dashboard de Administrador

- **Objetivo:** Centralizar a gestão da plataforma, incluindo produtos, utilizadores e conteúdos.
- **Elementos principais:**
 - **Estatísticas e gráficos:** Visão geral das vendas e métricas de utilização (RF.48, RF.49).
 - **Gestão de produtos e stock:** Ferramentas para adicionar, editar e monitorizar produtos (RF.11–RF.13).

- **Gestão de utilizadores:** Controlo de permissões e resolução de problemas (RF.44, RF.45).
- **Relatórios:** Geração de relatórios personalizáveis em PDF (RF.46, RF.47).
- **Notificações:** Alertas para questões críticas, como stock baixo ou tickets de suporte (RF.31).
- **Gestão do Blog:** Adicionar e editar publicações, interagindo diretamente com os utilizadores (RF.44).
- **Justificação:**
 - A estrutura hierárquica e os rótulos descritivos ("Estatísticas", "Gerir Produtos") garantem acesso rápido às funcionalidades mais usadas.
 - Navegação lateral permanente facilita a transição entre secções prioritárias.

Dashboard de Utilizador

- **Objetivo:** Fornecer informações centralizadas sobre as plantas do utilizador e promover a autonomia no cuidado diário.
- **Elementos principais:**
 - **Cards de plantas:** Exibição de informações como nome, estado de saúde e alertas (RF.22–RF.27).
 - **Monitorização em tempo real:** Gráficos detalhados de condições ambientais, como humidade e luz (RF.28–RF.30).
 - **Histórico de atividades:** Resumo das ações realizadas pelo utilizador, como regas ou fertilizações (RF.26).
 - **Notificações:** Lembretes automatizados para regar ou ajustar condições (RF.31, RF.32).
- **Justificação:**
 - O design *card-style* destaca informações críticas, permitindo que o utilizador atue rapidamente.
 - Navegação contextual nos *cards* simplifica a interação com dados detalhados.

Loja Online

- **Objetivo:** Oferecer uma experiência de compra otimizada e acessível.
- **Elementos principais:**
 - **Categorias de produtos:** Filtros e pesquisa por tipo, preço e popularidade (RF.1–RF.3).
 - **Carrinho de compras:** Visualização e gestão de itens selecionados (RF.4–RF.10).
 - **Página de checkout:** Inclusão de dados de pagamento e envio (RF.9, RF.10).
- **Justificação:**
 - A navegação local permite que os utilizadores encontrem produtos facilmente.
 - A interface responsiva assegura compatibilidade com dispositivos móveis.

Quiz

- **Objetivo:** Ajudar o utilizador a selecionar plantas ideais com base nas suas condições e necessidades.
- **Elementos principais:**
 - **Perguntas sequenciais:** O fluxo linear guia o utilizador para a melhor recomendação (RF.42, RF.43).
 - **Recomendações:** Apresentação de plantas compatíveis com ligação direta à loja online (RF.24, RF.43).
- **Justificação:**
 - O design *clean-style* elimina distrações, promovendo um foco total na tarefa.

Blog

- **Objetivo:** Educar e informar os utilizadores sobre jardinagem e práticas sustentáveis.
- **Elementos principais:**
 - **Artigos organizados por categorias:** Estrutura clara para navegação intuitiva.
 - **Integração com notificações:** Envio de alertas para novos conteúdos relevantes.
- **Justificação:**
 - Navegação global permite transição rápida entre o blog e outras secções.

Página Inicial

- **Objetivo:** Servir como ponto de entrada para todas as funcionalidades do site.
- **Elementos principais:**
 - **Links diretos:** Acesso imediato à loja, *dashboards*, quiz e blog.
 - **Elementos gráficos:** Destaque de funcionalidades principais e produtos populares.
- **Justificação:**
 - Organização lógica e menus globais asseguram que os utilizadores naveguem eficientemente.
- **Loja Online:** Organizada por categorias de produtos, o que facilita a filtragem e a navegação através de uma estrutura clara. Apresenta links para o Blog, permitindo que os utilizadores leiam artigos relacionados aos produtos.
- **Quiz:** Segue um fluxo linear, onde a navegação é sequencial, guiando o utilizador através de perguntas até ao resultado final. Após a conclusão, o utilizador pode ser redirecionado para plantas compatíveis disponíveis na Loja Online, dependendo do resultado.

Os diagramas de arquitetura de informação estão apresentados no **Anexo H** demonstram como cada secção da plataforma foi projetada para atender aos objetivos mencionados, detalhando a organização hierárquica e as conexões entre os diferentes módulos do sistema. Essa documentação serve como uma base sólida para futuras melhorias e expansões do projeto, assegurando que a navegação e a estrutura continuem eficientes e centradas nas necessidades dos utilizadores.

12.3. User Flow

O user-flow representa, de forma visual e sequencial, os caminhos que os utilizadores percorrem na aplicação para alcançar objetivos específicos. Este diagrama é uma aplicação prática da arquitetura da informação, destacando como os elementos estruturais e as funcionalidades da plataforma se conectam para proporcionar uma experiência fluida e eficiente.

No caso do **CultivUA**, o user-flow permite analisar e otimizar a navegação em funcionalidades como o registo de plantas, a gestão de kits ou a realização de compras na loja online. O diagrama de user-flow da aplicação encontra-se disponível no

, detalhando as interações chave e os fluxos de utilizador mais relevantes.

13. Diagrama MVC

O MVC (*Model-View-Controller*) descreve a arquitetura utilizada no desenvolvimento do sistema, que se baseia na divisão da aplicação em três componentes principais: *Model*, *View* e *Controller*. [14] Esta abordagem tem como objetivo separar as responsabilidades de processamento de dados, apresentação da interface e controlo da lógica de interação, o que permite facilitar a manutenção e escalabilidade do sistema. O *Model* é responsável pela gestão e manipulação dos dados, a *View* lida com a apresentação e interface com o utilizador, enquanto o *Controller* atua como intermediário, gerindo as interações entre a visualização e os dados, o que garante que as ações do utilizador sejam corretamente processadas e refletidas na aplicação.

Na figura seguinte é apresentado o diagrama MVC da plataforma **CultivUA**, caso seja necessário uma vista do diagrama com mais detalhe está disponível no **Anexo G**

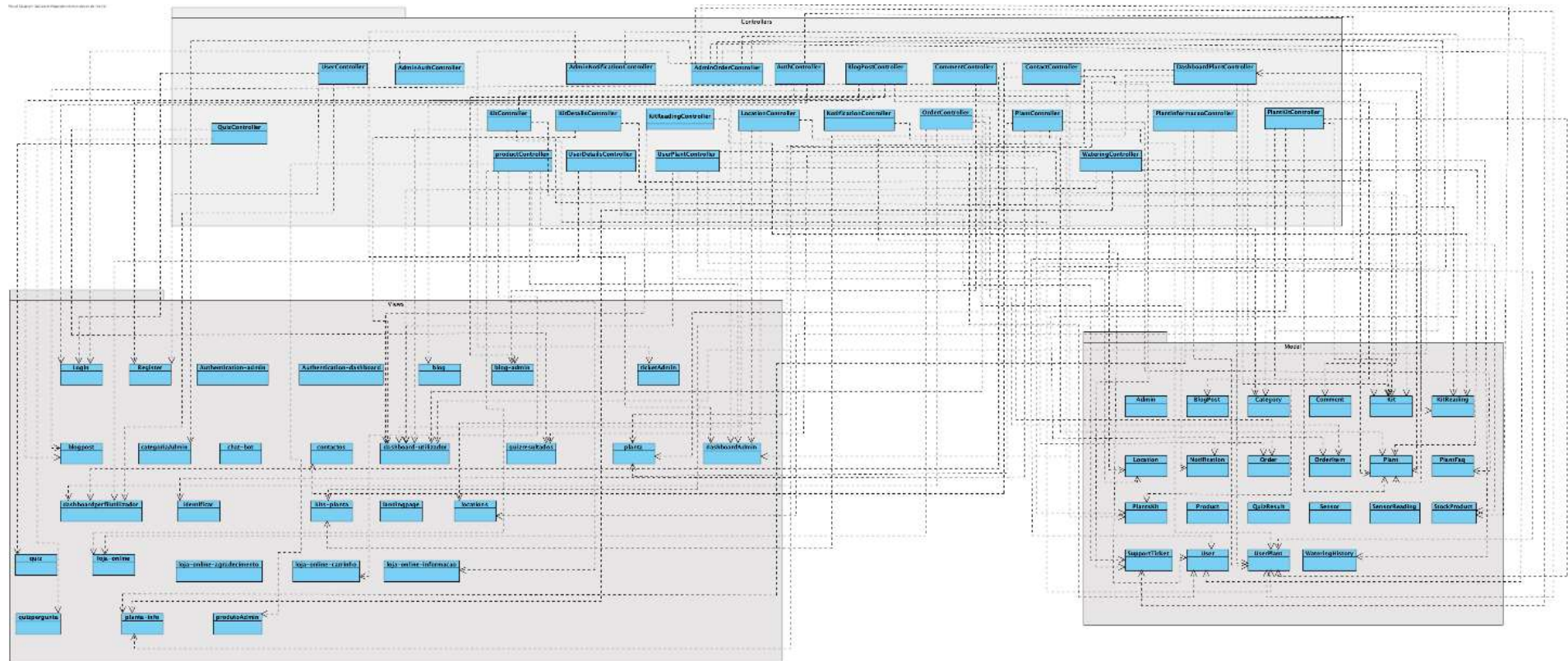


Figura 10 - Diagrama MVC

14. Diagrama de Arquitetura do Sistema

A imagem abaixo [Figura 11 - Diagrama de Arquitetura do Sistema], ilustra a arquitetura de uma aplicação web integrada com um sistema de hardware em Arduino. Esta arquitetura abrange a interação do utilizador através de uma interface web (*frontend*), a gestão dos dados (*backend*) e a comunicação com um dispositivo Arduino (Kit da Planta) para a recolha de dados.

A aplicação utiliza várias tecnologias, como o Angular para o *frontend*, Laravel para o *backend* e uma base de dados MySQL para armazenar informações. Além disso, a comunicação entre o Arduino e o servidor é realizada de forma wireless, com um script Python responsável pelo processamento dos dados entre a base de dados e o Arduino. A plataforma também comunica com duas APIs externas, a Perenual e a Plant ID, através do Laravel, para enriquecer as funcionalidades do sistema com dados adicionais.

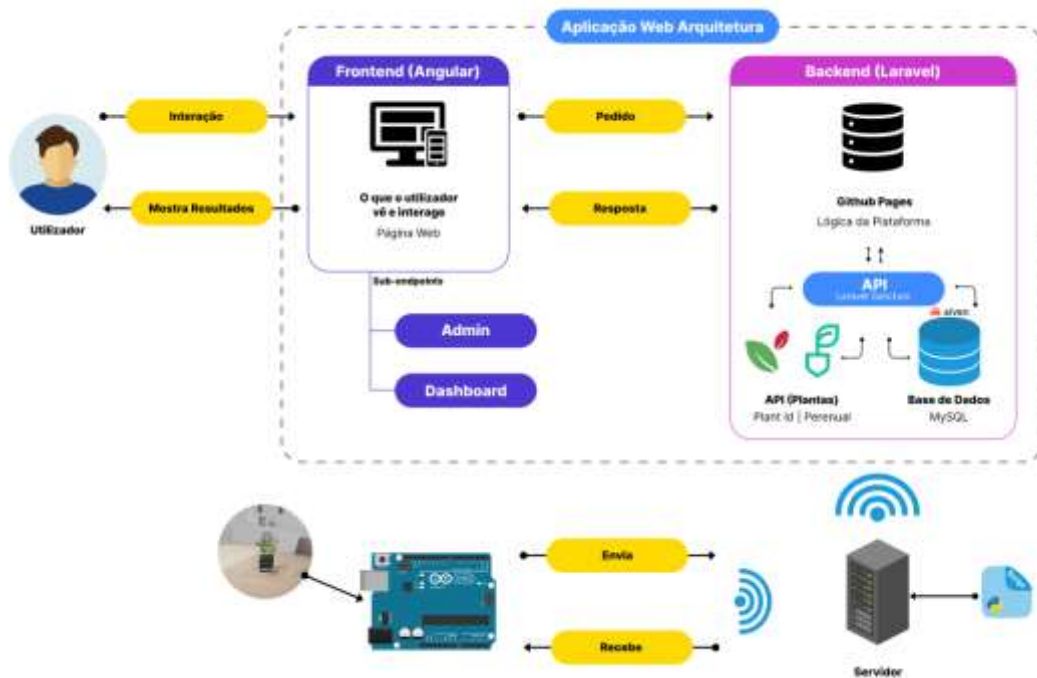


Figura 11 - Diagrama de Arquitetura do Sistema



15. Estratégia para Promover a Acessibilidade da Aplicação

A acessibilidade é um dos pilares fundamentais para garantir que o **CultivUA** seja inclusivo e facilmente utilizável por todos os utilizadores, incluindo aqueles com deficiência. O projeto adota uma abordagem abrangente que combina práticas de design inclusivo, conformidade com normas internacionais e uma experiência de utilização fluida e responsiva.

15.1. Princípios Gerais de Acessibilidade

- **Compatibilidade com WCAG 2.1:** A aplicação segue as Diretrizes de Acessibilidade para Conteúdo Web (WCAG 2.1), visando alcançar o nível AA. Isto assegura uma experiência de navegação inclusiva, capaz de atender às necessidades de utilizadores com diferentes limitações físicas, sensoriais ou cognitivas.
- **Estrutura Semântica e Navegação Clara:** O desenvolvimento do site segue uma hierarquia bem definida, com cabeçalhos, links e áreas de conteúdo organizados. Esta abordagem facilita a navegação lógica, beneficiando tanto os utilizadores de leitores de ecrã como aqueles que preferem uma estrutura intuitiva para encontrar informações.

15.2. Funcionalidades Específicas de Acessibilidade

- **Navegação por Teclado:** Todos os componentes da interface são acessíveis através do teclado, permitindo que utilizadores com mobilidade reduzida interajam com a plataforma sem necessidade de rato.
- **Texto Alternativo em Imagens:** Todas as imagens incluem descrições (alt text) adequadas, assegurando que utilizadores com deficiência visual compreendam o conteúdo visual apresentado.
- **Contraste e Redimensionamento de Texto:** A paleta de cores do site foi selecionada para garantir um contraste adequado entre texto e fundo. Além disso, a aplicação permite o ajuste do tamanho do texto, facilitando a leitura e a utilização para pessoas com limitações visuais.



15.3. Testes de Acessibilidade

- **Ferramentas de Avaliação:** Durante o desenvolvimento, ferramentas como o Google Lighthouse foram utilizadas para identificar e corrigir problemas de acessibilidade, garantindo conformidade com as normas estabelecidas.
- **Testes com Utilizadores Reais:** Foram realizados testes de usabilidade com a turma numa aula dedicada ao projeto, permitindo identificar áreas de melhoria e validar as funcionalidades de acessibilidade implementadas.

15.4. Usabilidade e Responsividade

Ao longo do desenvolvimento da plataforma, foi dada especial atenção à usabilidade e à responsividade da mesma, de forma a garantir que a plataforma atende às necessidades de todos os utilizadores, independentemente do dispositivo utilizado ou do seu nível de familiaridade com tecnologia. Estas características são essenciais para criar uma interface acessível e promover uma interação positiva e fluida com o sistema.

- **Usabilidade:** O website adota práticas de design inclusivo, permitindo que utilizadores com diferentes níveis de experiência tecnológica, desde iniciantes até avançados, naveguem facilmente na plataforma. A organização clara das informações e os sistemas de navegação intuitivos ajudam os utilizadores a encontrar rapidamente o que procuram, reduzindo obstáculos e promovendo uma experiência positiva.
- **Responsividade:** O design responsivo do **CultivUA** foi desenvolvido para oferecer uma experiência robusta em computadores, enquanto adaptações específicas foram implementadas para dispositivos móveis e tablets. O layout, os menus e os conteúdos ajustam-se automaticamente a diferentes resoluções, garantindo uma navegação fluida e eficiente em qualquer dispositivo. Elementos como botões, tabelas de dados e gráficos foram otimizados para ecrãs menores, mantendo a funcionalidade sem comprometer a experiência do utilizador. Esta abordagem garante que o **CultivUA** seja acessível, intuitivo e eficaz em qualquer contexto de utilização, promovendo a inclusão e a satisfação de todos os utilizadores.



16. Wireframes

Os *wireframes* são uma etapa essencial no processo de design de projetos digitais, como websites, aplicações e sistemas. Estes esboços simples representam a estrutura e o layout de uma interface de forma básica, sem se preocuparem com elementos visuais ou estilísticos. A sua principal vantagem é permitir que as ideias sejam exploradas de forma clara e eficiente, funcionando como um meio de comunicação entre os designers, programadores e outros membros da equipa. Por serem focados na organização e funcionalidade, ajudam a alinhar expectativas desde o início do projeto.

Outra vantagem dos *wireframes* de baixa fidelidade é o seu papel na criação de uma boa experiência do utilizador. Ao não incluir elementos visuais complexos, é possível concentrar-se inteiramente na usabilidade, nos fluxos de navegação e na organização das informações. Este foco permite identificar problemas ou pontos de melhoria logo numa fase inicial, poupando tempo e custos. Além disso, estes *wireframes* são rápidos de criar e fáceis de alterar, tornando o processo de iteração simples e eficaz. Assim, diferentes abordagens podem ser testadas, garantindo que o projeto responde às necessidades do público-alvo.

Aqui estão dois exemplos de *wireframes* para consulta. Os restantes podem ser encontrados na pasta do Onedrive disponível no **Anexo E**.

16.1. Landing Page

A página principal do website contém uma secção sobre a empresa e os produtos vendidos, apresenta alguns produtos, avaliações e publicações do blog.

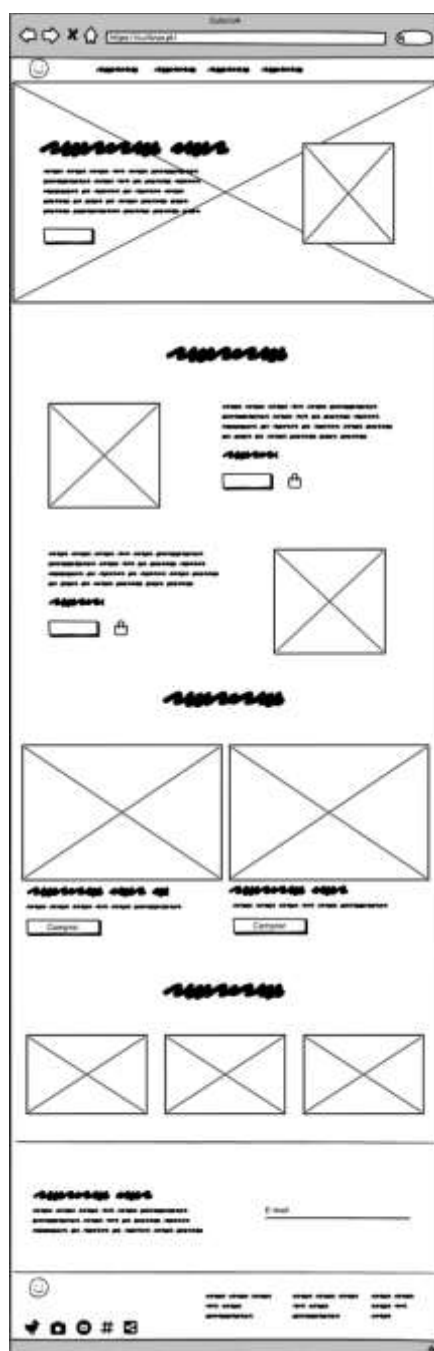


Figura 12 - Landing Page



16.2. Loja Online

A página da loja online tem como objetivo incentivar o utilizador a comprar produtos. Esta página exhibe os produtos mais vendidos, as novidades e as suas respetivas categorias. Permite ao utilizador pesquisar por categorias e aplicar filtros de preço e tamanho.

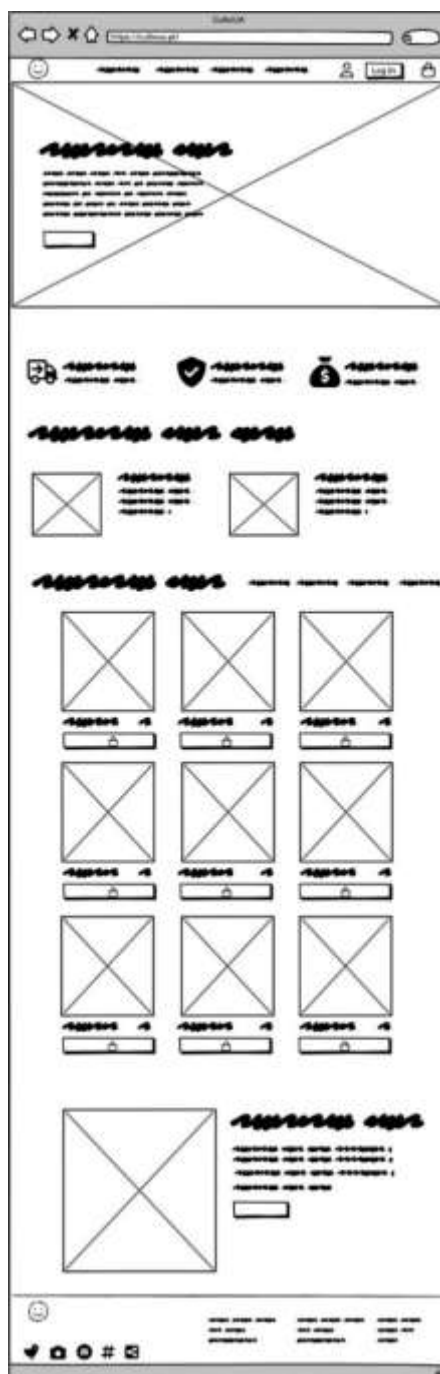


Figura 13 - Loja Online



17. Protótipo de Alta-fidelidade

Os *mockups* das páginas são representações visuais essenciais para compreender como o layout do site será estruturado e como os utilizadores irão interagir com as funcionalidades. Nesta apresentação, destacamos a landing page, projetada para oferecer uma experiência envolvente e intuitiva, e a dashboard do administrador, que foca na gestão eficiente de conteúdos e funcionalidades. Ambas as páginas se destacam pela organização clara das informações e pela facilidade de navegação.

Os *mockups* das restantes páginas, como a loja online, o blog e o quiz, estão disponíveis para consulta no projeto Figma, cujo link pode ser encontrado no

Anexo F.

17.1. Página Inicial (Landing Page)

A página inicial do **CultivUA** foi concebida para apresentar de forma clara e atraente o propósito da plataforma. Esta não mostra apenas as principais informações sobre a missão e os nossos valores, mas também convida o utilizador a explorar o que o site tem a oferecer. Com um design moderno e uma paleta de cores suaves onde o verde domina, a página inicial foca-se na sustentabilidade e no cultivo urbano, ao utilizar imagens naturais que transmitem a ideia de harmonia com o meio ambiente. As funcionalidades principais estão dispostas de maneira intuitiva, incluindo a exibição dos produtos populares, criando um ambiente acolhedor e funcional para qualquer utilizador que entre no site pela primeira vez.



Figura 14 - Mockup Landing Page



17.2. Dashboard do Admin

O painel de controlo do administrador oferece uma visão abrangente de todas as métricas importantes do site. Esta área é projetada para permitir que o administrador monitorize facilmente o desempenho do **CultivUA**, com gráficos interativos e módulos que fornecem dados sobre o número de utilizadores, vendas e produtos, permite também que sejam gerados relatórios sobre esses dados. A interface simples e organizada torna a gestão de informações eficiente. É um painel essencial para os administradores que necessitam de *insights* rápidos sobre a operação do site e o comportamento dos utilizadores, permitindo uma gestão eficiente e informada.

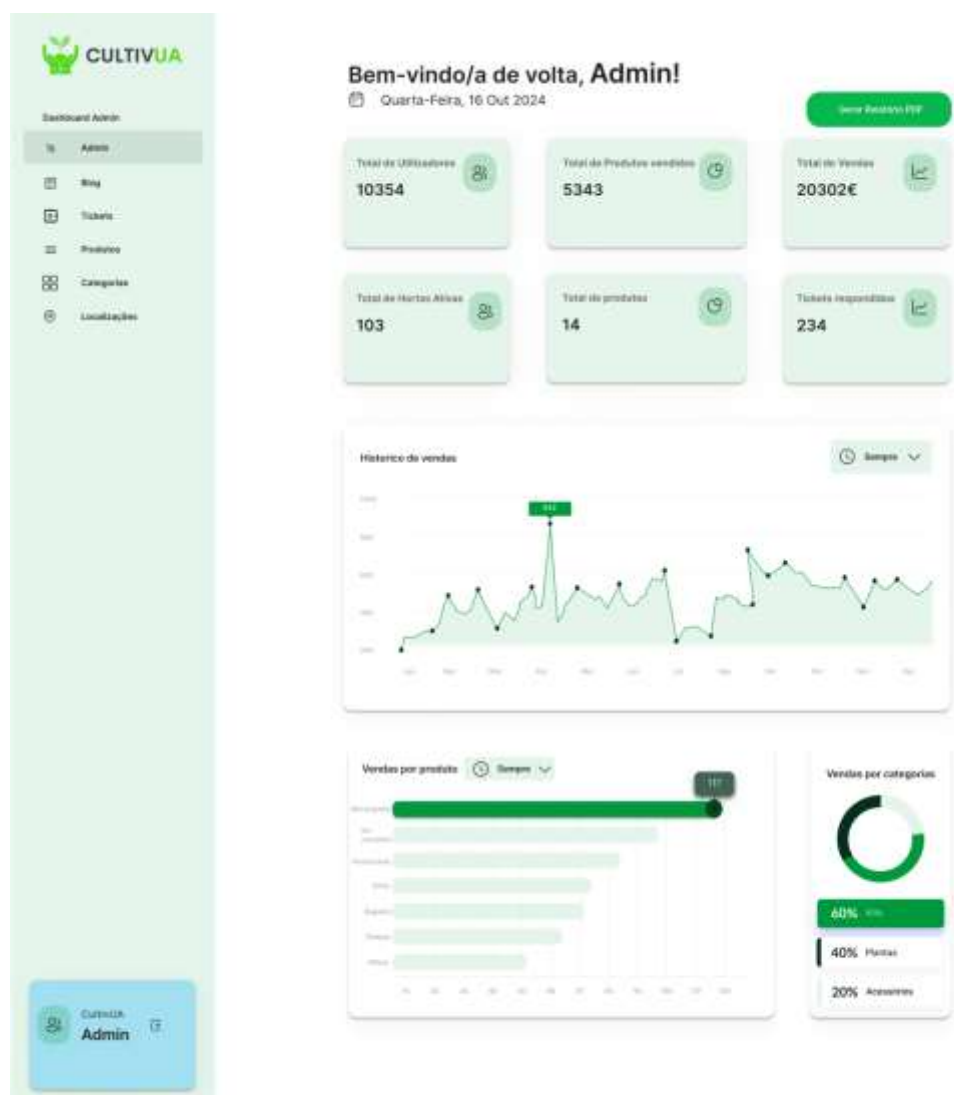


Figura 15 - Mockup Dashboard Admin



18. Demonstração do Website

Neste tópico, serão apresentadas algumas das principais funcionalidades da plataforma web já implementada.

Todas as funcionalidades e informações complementares estarão disponíveis no **Anexo K**, proporcionando uma visão completa do projeto.

18.1. Página Inicial (Landing Page)

A landing page do **CultivUA** destaca como a plataforma foi concebida para apresentar, de forma clara e apelativa, o seu propósito. A página introduz a missão e os valores do projeto, convidando os utilizadores a explorar o que o site tem para oferecer. Com um design moderno, uma paleta de cores suaves e imagens naturais, transmite a essência de sustentabilidade e cultivo urbano. Os elementos principais estão organizados de forma intuitiva, apresentando produtos populares e testemunhos de clientes, de forma a criar uma experiência acolhedora e envolvente para quem visita o site pela primeira vez.

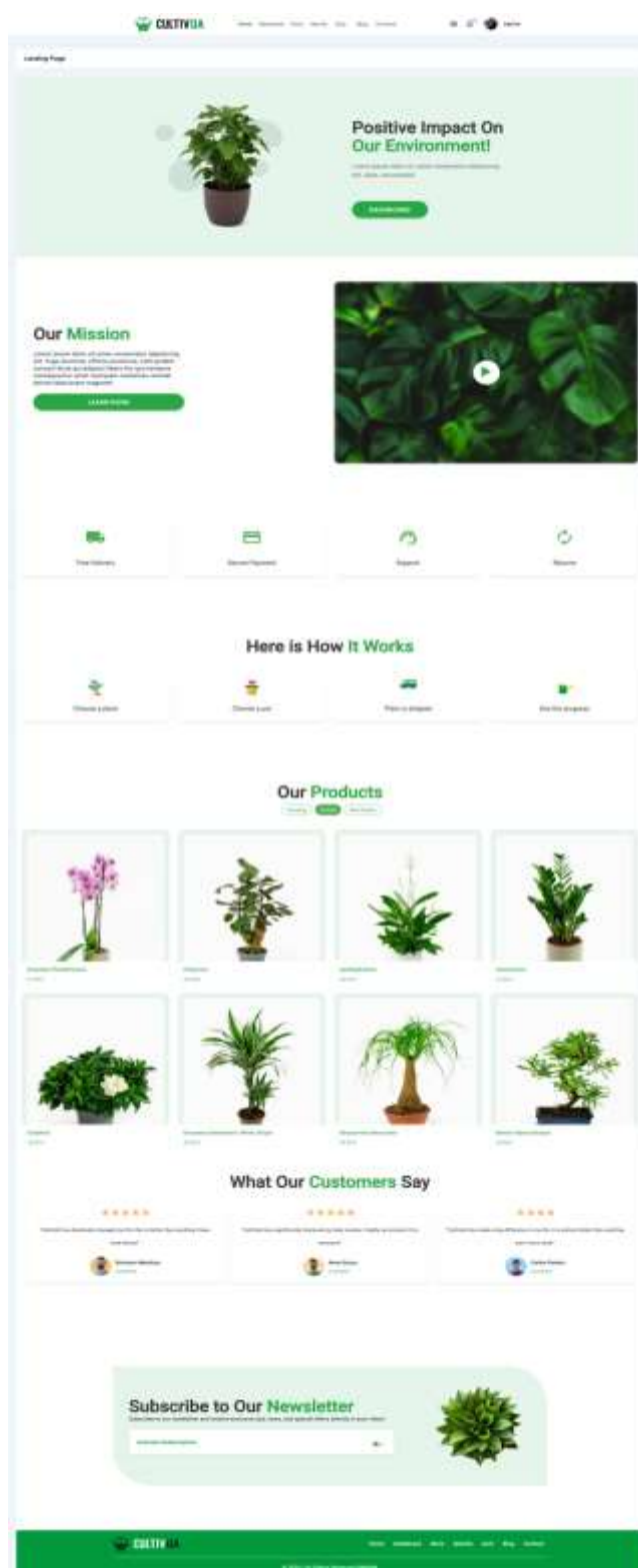


Figura 16 - Dashboard Landing Page

18.2. Dashboard do Admin

O painel de controlo do administrador oferece uma visão abrangente de todas as métricas importantes do site. Esta área é projetada para permitir que o administrador monitorize facilmente o desempenho do **CultivUA**, com gráficos interativos e módulos que fornecem dados sobre o número de utilizadores, vendas e produtos, permite também que sejam gerados relatórios sobre esses dados. A interface simples e organizada torna a gestão de informações eficiente, enquanto os filtros oferecem flexibilidade para personalizar as visualizações. É um painel essencial para os administradores que necessitam de *insights* rápidos sobre a operação do site e o comportamento dos utilizadores, permitindo uma gestão eficiente e informada.

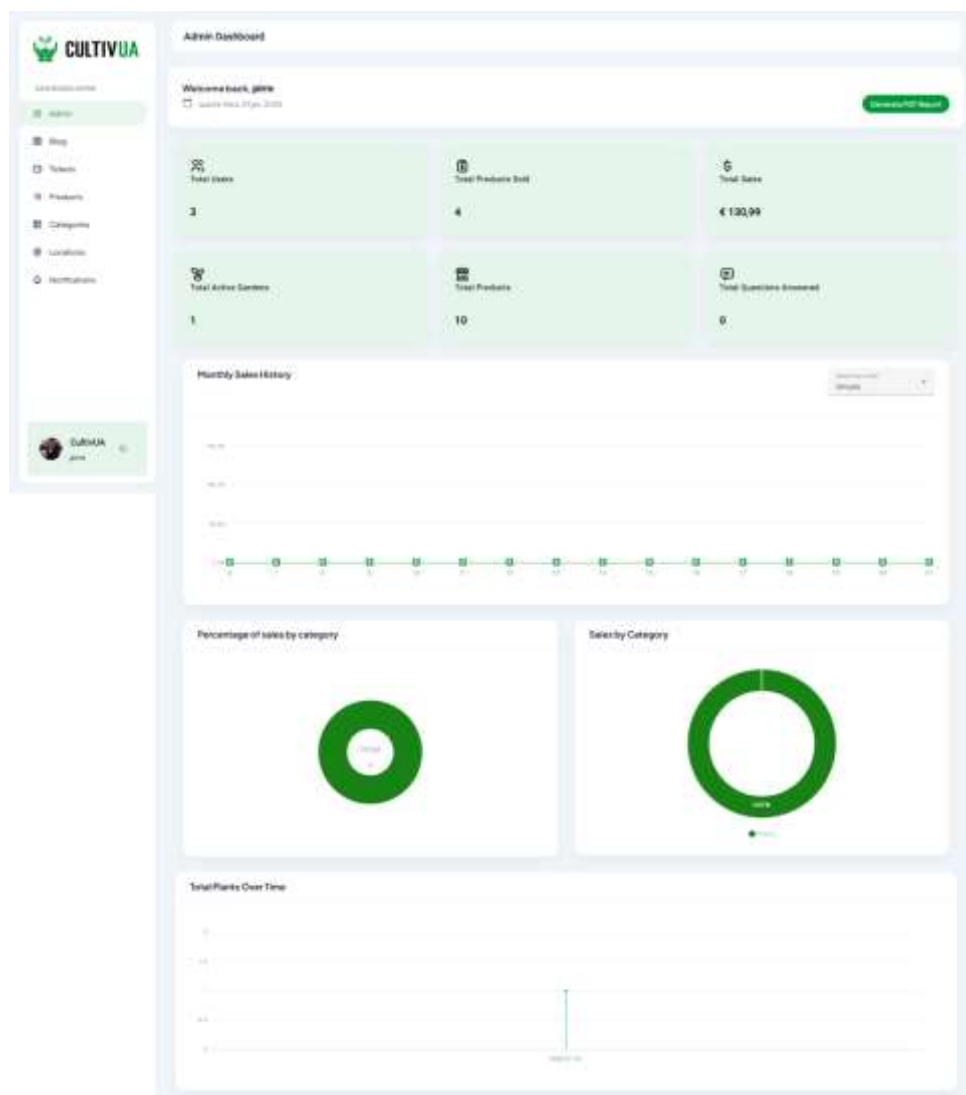


Figura 17 - Dashboard do Admin



18.3. Produtos

O painel de Produtos é a central para gestão e controlo do inventário no **CultivUA**. Ele permite ao administrador adicionar, editar ou remover produtos de forma intuitiva e rápida. Cada linha apresenta informações essenciais, como nome, categoria, preço e stock, garantindo uma visão clara do status dos itens disponíveis. Além disso, há opções de pesquisa e filtros para encontrar produtos específicos com facilidade. A interface organizada simplifica o processo de atualização, assegurando que o catálogo esteja sempre atualizado e alinhado às necessidades dos utilizadores.

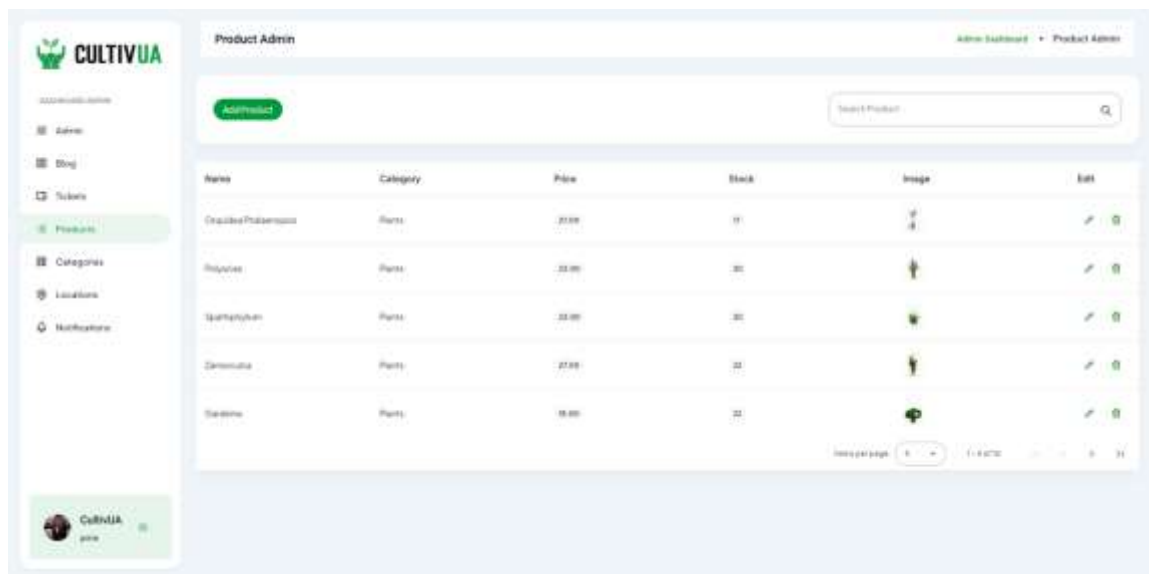


Figura 18 - Dashboard Produtos Admin



18.4. Categorias

No painel de Categorias, o administrador pode gerir as diferentes categorias de produtos disponíveis no site. Esta *dashboard* possibilita a criação, edição ou exclusão de categorias, bem como a visualização de quantos produtos estão associados a cada uma delas. A interface é clara e funcional, com controlos acessíveis que permitem a sua organização de forma eficiente. Esse painel é essencial para manter uma estrutura organizada no site, facilitando a navegação para os utilizadores e otimizando a procura por produtos.

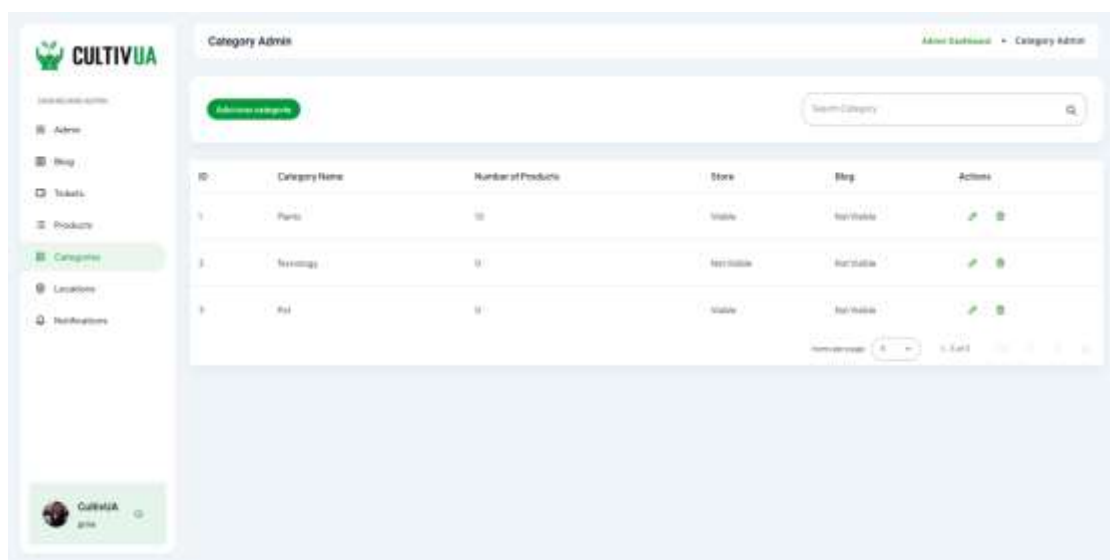


Figura 19 - Dashboard Categorias Admin



18.5. Blog Admin

O painel Blog Admin é o espaço dedicado à gestão de publicações do blog do **CultivUA**. Aqui, o administrador pode criar novos conteúdos, editar ou remover publicações existentes assim como os seus comentários, além de categorizá-las para melhor organização. O formulário de criação é simples e eficiente, com campos para título, descrição e imagem, permitindo que o conteúdo seja personalizado e visualmente atraente. A funcionalidade de pesquisa facilita a gestão de publicações, garantindo que as informações sejam relevantes e atualizadas para os utilizadores.

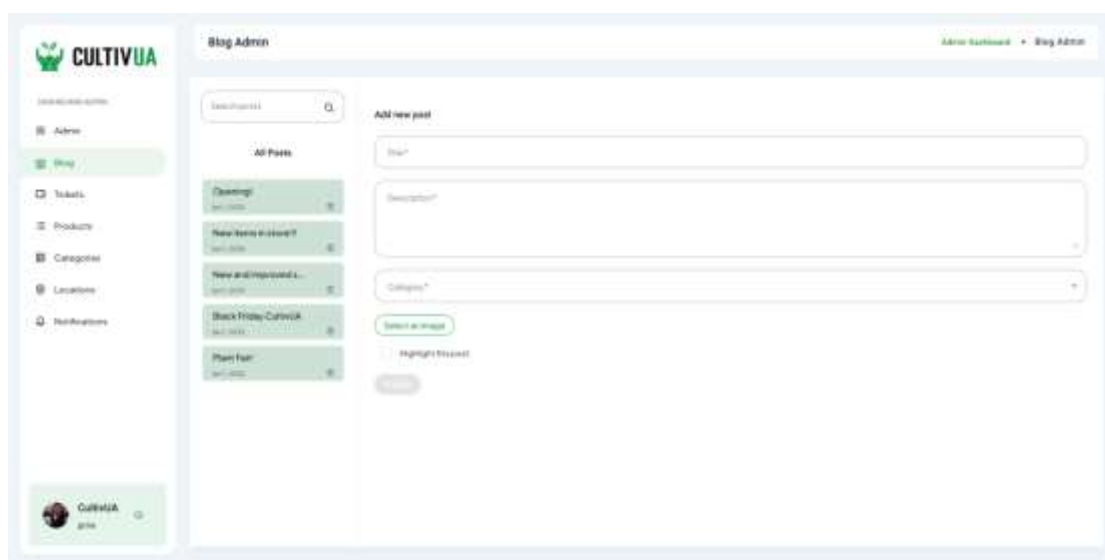


Figura 20 - Dashboard Blog Admin



18.6. Tickets

O painel de Tickets é onde o administrador pode acompanhar e resolver questões levantadas pelos utilizadores. Este exibe uma lista organizada de tickets, com informações como o nome do utilizador, status (aberto ou fechado) e data de criação. Este módulo oferece ferramentas para visualizar detalhes, responder diretamente ou fechar tickets resolvidos. Com uma interface clara e recursos fáceis de usar, o painel de tickets assegura uma comunicação eficiente e um suporte ao cliente de alta qualidade.

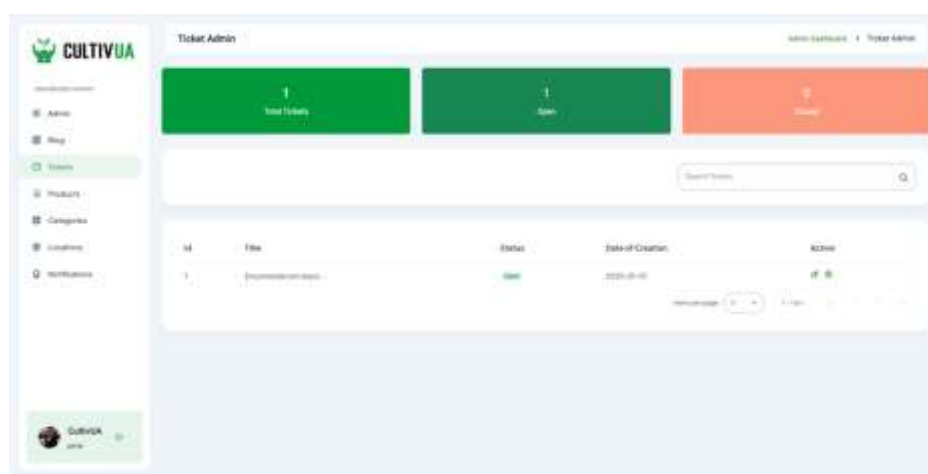


Figura 21 - Dashboard Tickets Admin

18.7. *Dashboard do Utilizador*

O painel do utilizador foi desenvolvido para proporcionar uma experiência personalizada, focando no acompanhamento do estado das plantas e nas necessidades do utilizador. A interface limpa e agradável permite ao cliente monitorizar facilmente a saúde das suas plantas, com gráficos sobre humidade, temperatura e consumo de água, assim como o estado geral da horta urbana na opção de ver mais. A navegação é intuitiva, permitindo que o utilizador aceda rapidamente às suas plantas, visualize os detalhes de cada uma e adicione novas. Com uma abordagem focada no cuidado das plantas, o painel proporciona a experiência ideal para quem deseja manter uma horta urbana saudável, de forma prática e acessível.

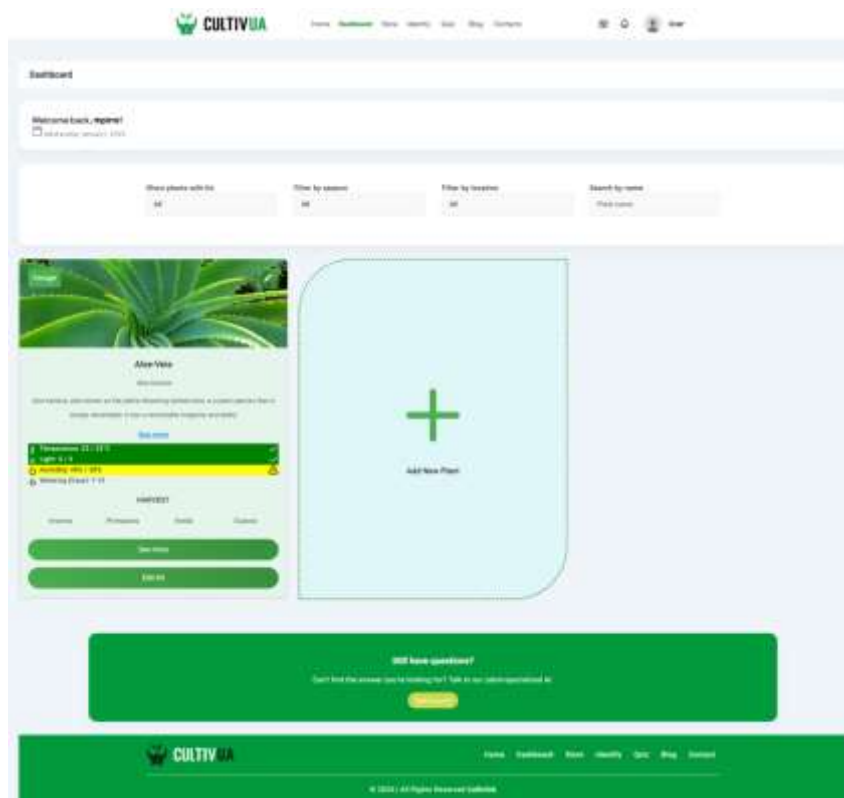
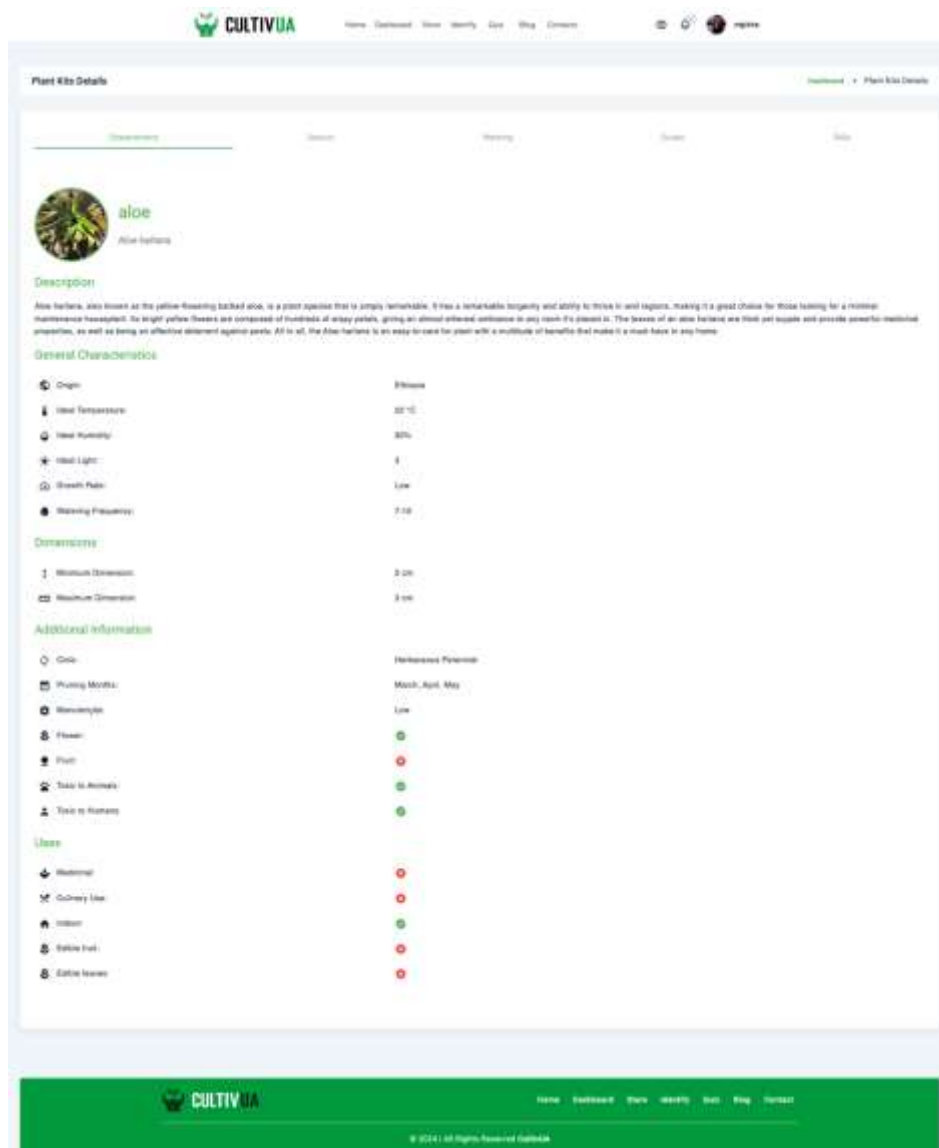


Figura 22 - *Dashboard do Cliente*

18.8. Detalhes da Planta

O painel de detalhes da planta oferece informações detalhadas sobre o estado de uma planta específica. Os utilizadores podem visualizar métricas como saúde, dimensões, e histórico de rega. Este painel é essencial para quem procura monitorizar cada detalhe e garantir que as suas plantas prosperem.



The screenshot displays the 'Plant Kit Details' page for 'Aloe Vera' on the CultivUA platform. The page is structured with a navigation bar at the top, a main content area with tabs for Overview, Details, History, and Settings, and a footer. The 'Details' tab is active, showing a comprehensive overview of the plant's characteristics and care requirements.

Plant Kit Details

Overview | Details | History | Settings

Aloe Vera

Description

Aloe Vera, also known as the yellow-flowering aloe, is a plant species that is simply remarkable. It has a remarkable longevity and ability to thrive in arid regions, making it a great choice for those looking for a resilient, low-maintenance houseplant. Its large, yellow flowers are composed of hundreds of tiny petals, giving an almost ethereal appearance in any room it's placed in. The leaves of an aloe vera plant are thick and succulent and provide powerful medicinal properties, as well as being an effective natural skin care product. All in all, the Aloe Vera is an easy-to-care-for plant with a multitude of benefits that makes it a must-have in any home.

General Characteristics

Characteristic	Value
Origin	Brazil
Ideal Temperature	22-30°C
Ideal Humidity	30%
Ideal Light	4
Growth Rate	Low
Watering Frequency	2-3x

Dimensions

Measurement	Value
Maximum Dimension	2.2m
Minimum Dimension	3 cm

Additional Information

Category	Value
Color	Herbaceous Perennial
Planting Month	March, April, May
Reproduction	Low
Flower	Yes
Fruit	No
Toxic to Animals	No
Toxic to Humans	No

Uses

Use	Value
Medicinal	No
Culinary Use	No
Ornamental	Yes
Edible Fruit	No
Edible Leaves	No

CULTIVUA

Home | Dashboard | Store | My Plants | Settings | Help | Contact

© 2024 | All Rights Reserved | CultivUA

Figura 23 - Detalhes Planta (Características)

18.9. Loja Online

A loja online do **CultivUA** oferece uma vasta seleção de produtos relacionados com jardinagem e cultivo. Com categorias organizadas e filtros de pesquisa, os utilizadores podem encontrar rapidamente o que precisam. A experiência de compra é fluida, permitindo adicionar itens ao carrinho com um clique e finalizar a compra de forma prática.

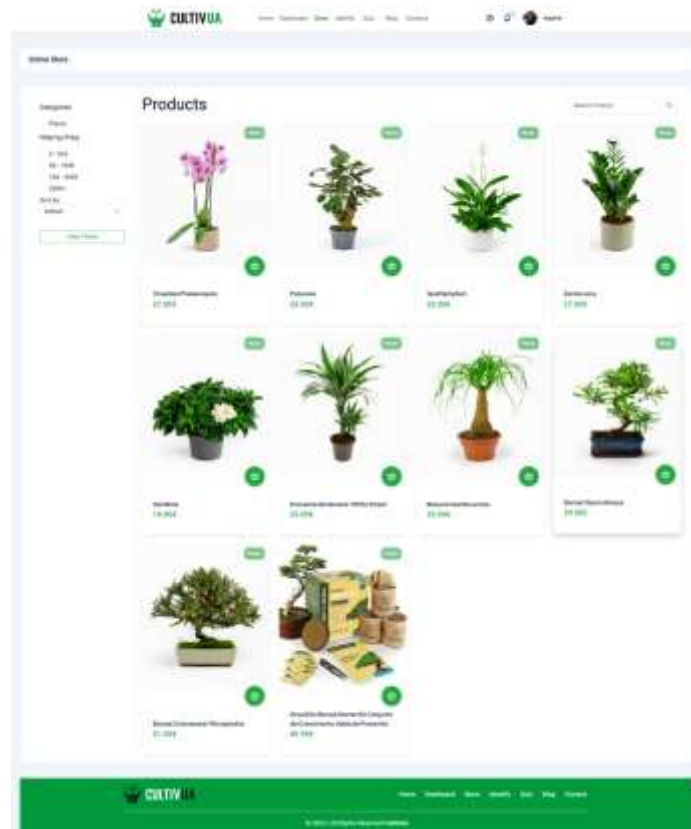


Figura 24 - Mockup Loja Online



Checkout

Nesta etapa, o utilizador pode seleccionar e preencher os dados necessários para o pagamento e para o envio dos produtos, através do Stripe. É possível escolher entre diferentes métodos de entrega. A interface é simples, garantindo que o processo de inserção de moradas e opções de transporte seja rápido e eficiente.

The screenshot displays a checkout interface. On the left, a green sidebar shows the total amount: "Organized Photoalbums €27,00". The main area contains a form titled "Dados de envio" (Shipping Data) with fields for "E-mail", "Endereço de entrega" (Delivery Address), "País" (Country) set to "Portugal", "Código postal" (Postal Code), and "Cidade" (City). Below this is the "Dados de pagamento" (Payment Data) section, which includes a "Número de cartão" (Card Number) field, a "Data de validade" (Expiration Date) field, and a "Nome do titular" (Cardholder Name) field. A checkbox indicates "O meu método de pagamento está guardado para o próximo" (My payment method is saved for next time). A green "Pagar" (Pay) button is at the bottom. A footer bar contains links for "Política de Privacidade", "Termos", and "Ajuda".

Figura 25 - Checkout

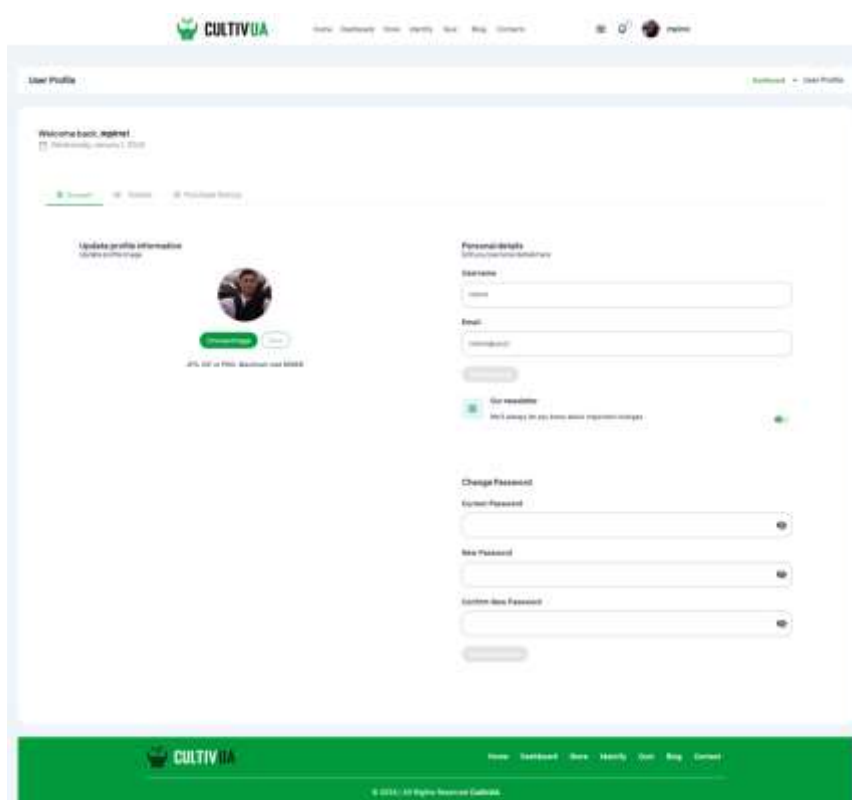


18.10. Loja Online – Compra de Kit

Na plataforma web, o processo de compra varia consoante o tipo de produto adquirido. Para produtos individuais, como plantas, vasos ou sementes, o cliente efetua o pagamento através da Stripe, sendo-lhe entregue exclusivamente os itens encomendados. No caso da aquisição de um kit, embora o pagamento siga o mesmo processo, a encomenda é acompanhada por dois itens adicionais: um cartão com uma mensagem personalizada de agradecimento, contendo também o número do kit necessário para a sua ativação, e um folheto informativo com as instruções detalhadas para a instalação do kit e a respetiva integração com a plataforma web. Estes podem ser visualizados no **Anexo L**.

18.11. Gestão da Conta

O painel de gestão da conta permite ao utilizador visualizar e editar as informações pessoais associadas à sua conta. Aqui, é possível alterar dados como nome, endereço de e-mail e palavra-passe. A interface é simples, garantindo que qualquer atualização seja feita de forma rápida e fácil. Assegura que os utilizadores possam manter as suas informações atualizadas e personalizadas de acordo com as suas necessidades.



The screenshot displays the 'User Profile' page of the Cultivua application. The page is divided into two main sections: a left sidebar and a right main content area. The sidebar contains a 'Welcome back, Miguel' message, a date 'Wednesday, January 1, 2020', and a 'Update profile information' button. The main content area is titled 'Personal details' and includes fields for 'Username' (containing 'miguel'), 'Email' (containing 'miguel@cultivua.pt'), and a 'Get newsletter' checkbox. Below these fields is a 'Change Password' section with three input fields: 'Current Password', 'New Password', and 'Confirm New Password'. The page features a green header and footer with the Cultivua logo and navigation links.

Figura 26 - Perfil Utilizador

18.12. Contactos

A página de contactos é uma secção importantesdo site, pois oferece informação sobre quem somos e a nossa missão. Introduce também a equipa aos utilizadores e permite-os entrar em contacto. O design foi elaborado para ser simples e direto, com um formulário de contacto intuitivo, informações de localização, e opções de comunicação, como e-mail e telefone. O layout *clean* permite que o utilizador rapidamente encontre a melhor forma de resolver as suas dúvidas ou obter apoio. A experiência é otimizada para ser rápida e eficiente, facilitando a resolução de questões.

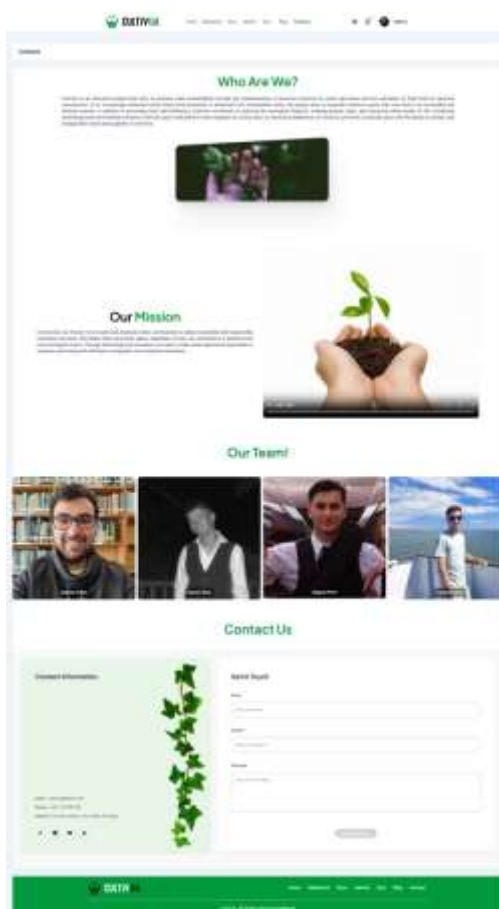


Figura 27 - Contactos

18.13. Quiz

O quiz do **CultivUA** foi projetado para guiar o utilizador de forma interativa, ajudando-o a identificar as melhores opções de plantas para o seu espaço e necessidades. Com um fluxo de navegação simples e sequencial, o quiz conduz o utilizador através de perguntas sobre preferências e condições ambientais, como luz disponível e tempo para cuidados.

A interface é intuitiva e agradável, utilizando ícones e textos claros para facilitar a compreensão. O resultado oferece recomendações personalizadas de plantas, tornando a experiência do utilizador mais envolvente e educativa.



Figura 28 - Quiz

18.14. Identificar Planta

A funcionalidade de identificação de plantas do **CultivUA** foi criada para oferecer aos utilizadores uma maneira simples e eficiente de descobrir mais sobre as plantas que encontram.

Com esta ferramenta, basta carregar uma fotografia da planta desejada e deixar que a tecnologia de identificação faça o resto. O sistema analisa a imagem, identificando automaticamente a espécie e apresentando informações detalhadas sobre ela.

A interface é amigável e minimalista, permitindo que os utilizadores façam o upload de imagens de forma rápida e intuitiva. Após a identificação, o utilizador recebe informações algumas informações sobre a planta e a opção de ir para a loja ou de a adicionar à sua *dashboard* pessoal.

Esta ferramenta não só é educativa, mas também prática, ideal para quem deseja expandir o conhecimento sobre as plantas ao seu redor ou iniciar novas aventuras no mundo da jardinagem.



Figura 29 - Identificar planta

18.15. Chatbot – Plant Lover

A funcionalidade de *chatbot* é uma ferramenta interativa projetada para ajudar os utilizadores a obterem respostas rápidas e personalizadas sobre o cuidado ou qualquer outra dúvida sobre plantas. Alimentado por inteligência artificial, este assistente virtual oferece sugestões, dicas práticas e responde a perguntas como “Quais são as melhores plantas para criar com pouco espaço?” ou “Como cuidar de morangos?”.

Com uma interface simples e amigável, o utilizador pode enviar mensagens diretamente para o *chatbot* e receber orientações claras e úteis. É uma solução prática para quem procura apoio instantâneo na manutenção do seu jardim ou está à procura de novas ideias para cultivo. O “Plant Lover” é o seu parceiro ideal para uma experiência de jardinagem ainda mais rica e produtiva.



Figura 30 - ChatBot

19. Testes e Validação

No âmbito do desenvolvimento do projeto **CultivUA**, foi dada especial atenção à realização de testes e validação, fundamentais para garantir a qualidade, robustez e eficiência das soluções implementadas. Dado o carácter tecnológico e inovador do projeto, foi crucial assegurar que tanto a plataforma web como os seus componentes funcionam de forma adequada e consistente em diferentes cenários de utilização.

Nesta secção, descrevem-se os testes realizados nas diferentes camadas da aplicação, abrangendo os serviços e componentes desenvolvidos em Angular e os controladores da aplicação em Laravel. Foram utilizados *frameworks* e ferramentas específicas para a execução dos testes, tais como Jasmine[15] e Karma[16] para Angular, Pest[17] para Laravel, e o Google Lighthouse[18] para avaliar métricas de desempenho, acessibilidade, SEO (*Search Engine Optimization*) e boas práticas. Este processo permitiu identificar e corrigir potenciais problemas, garantindo uma experiência otimizada para os utilizadores finais e promovendo a fiabilidade da plataforma.

Durante o desenvolvimento de software, os testes de sistema desempenham um papel crucial, uma vez que são realizados no sistema como um todo para verificar se este satisfaz todos os requisitos especificados.

19.1. Testes de Sistema e Validação

Os testes de sistema constituem uma etapa essencial no ciclo de desenvolvimento de software, permitindo verificar o comportamento do sistema como um todo e assegurar que este satisfaz os requisitos especificados. Esta fase é particularmente relevante para validar a integração de todos os componentes e garantir que a plataforma responde às expectativas dos utilizadores em termos funcionais e não funcionais.

No âmbito do projeto **CultivUA**, os testes de sistema foram realizados de forma colaborativa pelos membros do grupo, especialmente nas fases finais do desenvolvimento. Estes testes focaram-se em avaliar a funcionalidade global da plataforma web, incluindo a interação entre os diferentes módulos, a experiência do utilizador e a conformidade com os objetivos do projeto.

Durante a realização dos testes, foram simulados cenários reais de utilização para verificar o comportamento da plataforma em situações diversas, desde o acesso à loja online e à *dashboard* interativa até à integração com os sensores IoT. Desta forma, foi possível identificar inconsistências e implementar as devidas correções, garantindo que a plataforma não apenas funcionasse conforme o esperado, mas também proporcionasse uma experiência intuitiva e satisfatória para os utilizadores finais.

Os resultados positivos desta fase demonstraram a capacidade da plataforma em atender às necessidades do público-alvo, evidenciando a eficácia do processo de desenvolvimento e validação adotado no projeto **CultivUA**.

19.2. Testes Unitários

Os testes unitários representam uma etapa essencial no desenvolvimento de software, focando-se na validação de componentes ou funcionalidades isoladas do sistema. Este tipo de teste permite identificar problemas em módulos específicos, garantindo a sua correta implementação antes da integração com outras partes do sistema. No projeto **CultivUA**, foi dado ênfase especial à realização de testes unitários, de forma a assegurar a qualidade e a confiabilidade tanto da aplicação *front-end*, desenvolvida em Angular, como da aplicação *back-end*, implementada em Laravel.

Para os testes dos serviços e componentes desenvolvidos em Angular, foram utilizadas as ferramentas Jasmine e Karma. O Jasmine é um framework para testes de comportamento em JavaScript que permite escrever testes claros e estruturados com uma sintaxe simples e expressiva. Com esta ferramenta, foram criadas especificações de teste que verificaram o correto funcionamento de funcionalidades individuais, como validações de formulários, serviços de comunicação com a API e componentes visuais da interface. Já o Karma, por sua vez, é um *test runner* que executa os testes em múltiplos ambientes, como diferentes navegadores, automatizando a execução dos testes escritos com Jasmine e fornecendo relatórios detalhados sobre o progresso e os resultados obtidos.

No caso do *back-end*, implementado em Laravel, os testes unitários foram realizados com o uso do Pest, um *framework* moderno e minimalista para testes em PHP. Desenvolvido com foco em simplicidade e desempenho, o Pest oferece uma sintaxe mais limpa e legível, permitindo que os testes sejam escritos de forma concisa e intuitiva. Esta ferramenta foi utilizada para validar os *controllers* e a lógica da aplicação Laravel, garantindo que operações críticas, como a gestão de utilizadores e o processamento de pedidos, fossem realizadas de forma correta e eficiente.

A realização dos testes unitários durante o desenvolvimento do **CultivUA** foi crucial para identificar e corrigir erros, reduzindo significativamente os custos associados a problemas encontrados nas fases posteriores. Ao garantir a robustez de cada módulo individual, os testes unitários facilitaram o processo de integração e contribuíram para a construção de um sistema mais estável e confiável. A utilização de ferramentas reconhecidas, como Jasmine, Karma e Pest, estabeleceu uma base sólida para a validação da plataforma, assegurando que cada componente cumprisse os seus objetivos funcionais e operasse de forma consistente em diversos cenários de utilização. Esta abordagem reforçou a qualidade geral do software e promoveu a confiança na solução apresentada pelo **CultivUA**.

19.2.1. Testes Unitários no Angular

Como referido anteriormente, os testes unitários em Angular, foram realizados tanto nos serviços quanto nos componentes, sendo que cada tipo de teste se foca em diferentes aspetos do funcionamento da aplicação. Os testes de serviços concentram-se em validar a lógica e o comportamento de funcionalidades específicas, geralmente independentes da interface do utilizador. Por outro lado, os testes de componentes avaliam elementos da interface e interações com o utilizador, como validações de formulários e reações a eventos. Estas diferenças refletem-se tanto nos objetivos dos testes quanto na forma como são implementados.

No caso dos serviços, um exemplo de teste unitário foi realizado no serviço do carrinho de compras. Como se pode verificar na figura a seguir, este teste verifica se um item é corretamente adicionado ao carrinho de compras. Para tal, foi criado um item de exemplo e chamado o método **addToCart** do serviço. Em seguida, verificou-se o estado interno do carrinho para assegurar que o item foi corretamente adicionado. Este tipo de teste é especialmente útil para garantir que a lógica de manipulação de dados no serviço funciona como esperado, independentemente da interface. No final, o teste apresentou um resultado positivo, confirmando que o comportamento do serviço está alinhado com os requisitos definidos.

```
it('should add an item to the cart', () => {  
  const product: CartItem = { id: 1, name: 'Product 123', price: 100, quantity: 1 };  
  service.addToCart(product);  
  
  const cart = service['cartSubject'].getValue();  
  expect(cart.length).toBe(1);  
  expect(cart[0]).toEqual(product);  
});
```

Figura 31 - Teste unitário ao service do carrinho de compras


```
CartService
  should be created
  should add an item to the cart
  should remove an item from the cart
  should update the quantity of an existing item in the cart
```

Figura 32 - Resultado do teste do service do carrinho de compras

Por outro lado, nos componentes, o foco recai sobre a interação entre a interface do utilizador e os dados. Um exemplo disso, demonstrado na figura a seguir, foi o teste unitário realizado no componente de login da plataforma. Este teste avaliou a validação do controlo de palavra-passe no formulário, e verifica se o campo foi configurado como obrigatório. Para isso, o teste definiu o valor do campo como vazio e depois como uma palavra-passe válida, e verifica se a propriedade *valid* respondia corretamente às mudanças. Este tipo de teste assegura que a lógica de validação implementada no componente funciona como esperado, melhorando a usabilidade e a experiência do utilizador. No final, este teste também foi bem-sucedido, reforçando a confiança na funcionalidade da aplicação.

```
it('should make password control required', () => {
  const passwordControl = component.form.get('password');
  if (passwordControl) {
    passwordControl.setValue('');
    expect(passwordControl.valid).toBeFalsy();

    passwordControl.setValue('password');
    expect(passwordControl.valid).toBeTruthy();
  }
});
```

Figura 33 - Teste unitário á componente login da plataforma

```
AppSideLoginComponent
  should create
  should make password control required
  should have a form with email and password controls
  should make email control required and validate as email
```

Figura 34 - Resultado dos testes da componente de login da plataforma

Os testes realizados nos serviços e nos componentes são complementares, permitindo verificar tanto a lógica de negócio quanto as interações do utilizador com a interface. Esta abordagem garante uma cobertura abrangente e uma maior confiança na qualidade do software desenvolvido. Todos os testes unitários criados para a aplicação Angular do **CultivUA** estão disponíveis no **Anexo D**, para consulta ou verificação caso necessário.

19.2.1. Testes Unitários em Laravel

Os testes unitários em Laravel desempenham um papel crucial na validação da lógica e na garantia de que os *controllers* e outros componentes funcionam corretamente. Ao contrário dos testes realizados em Angular, que se concentram tanto na lógica implementada nos serviços como nas interações e validações na interface do utilizador, os testes em Laravel têm um foco mais aprofundado na camada de *back-end*, incluindo a interação com a base de dados, a autenticação e os *endpoints* da API.

Enquanto os testes em Angular verificam, por exemplo, a correta validação de formulários ou a manipulação de dados em serviços, os testes em Laravel avaliam aspetos como a criação e atualização de recursos, a proteção de rotas e o cumprimento de regras. Essa diferença de foco reflete as responsabilidades específicas de cada tecnologia no contexto do projeto.

No exemplo da figura abaixo, é apresentado um teste realizado em Laravel utilizando o Pest. Este teste verifica a criação de um ticket de suporte:

```
it('cria um ticket de suporte com sucesso', function () { void {  
    // Cria um utilizador para autenticação  
    $user = User::factory()->create();  
  
    // Autenticar o utilizador com Sanctum  
    \Laravel\Sanctum\Sanctum::actingAs($user);  
  
    $data = [  
        'subject' => 'Problema com a conta',  
        'message' => 'Estou a ter dificuldades para aceder à minha conta.',  
    ];  
  
    $response = $this->postJson('/api/ticket-user', $data);  
  
    // Verificar se o status é 201 (Created)  
    $response->assertStatus(201)  
        ->assertJson([  
            'message' => 'Ticket criado com sucesso!',  
            'ticket' => [  
                'subject' => 'Problema com a conta',  
                'message' => 'Estou a ter dificuldades para aceder à minha conta.',  
            ],  
        ]);  
  
    // Verificar se o ticket foi realmente criado na base de dados  
    $this->assertDatabaseHas('support_tickets', [  
        'subject' => 'Problema com a conta',  
        'message' => 'Estou a ter dificuldades para aceder à minha conta.',  
        'user_id' => $user->id,  
    ]);  
});
```

Figura 35 - Teste unitário ao controller responsável pela criação de Tickets

Este teste começa por criar um utilizador fictício utilizando a funcionalidade de *factories* do Laravel. Em seguida, o utilizador é autenticado através do Sanctum, uma biblioteca para autenticação de API. Com o utilizador autenticado, é enviada uma requisição POST para a rota responsável pela criação de tickets, fornecendo os dados necessários.

O teste verifica se a resposta da API devolve o status 201 (*Created*) e se o conteúdo JSON da resposta contém a mensagem esperada e os dados do ticket criado. Por fim, é utilizada a função `assertDatabaseHas` para confirmar que o ticket foi efetivamente armazenado na base de dados com os valores corretos, garantindo a integridade da operação.

Este exemplo ilustra a robustez dos testes em Laravel, que não apenas validam a resposta da API mas também verificam a persistência de dados, assegurando que as operações no back-end são realizadas corretamente. No final, o teste apresenta um resultado positivo, indicando que todas as etapas foram bem-sucedidas e que a funcionalidade está alinhada com os requisitos esperados.

```
PASS Tests\Feature\ContactControllerTest
✓ it cria um ticket de suporte com sucesso
✓ it retorna erro ao tentar criar ticket sem estar autenticado
✓ it retorna erro ao criar ticket com dados inválidos
```

Figura 36 - Resultado do teste ao controller de contactos

Os testes unitários realizados em Laravel foram fundamentais para assegurar a fiabilidade dos controladores e a conformidade com os requisitos funcionais da aplicação. Para consulta, todos os testes unitários desenvolvidos em Laravel estão disponíveis no **Anexo D**, permitindo a verificação conforme necessário.

19.3. Testes de qualidade com o Lighthouse

O Lighthouse é uma ferramenta de código aberto desenvolvida pela Google para avaliar a qualidade de aplicações web. Este recurso realiza auditorias automáticas em vários aspetos de uma aplicação, fornecendo relatórios detalhados sobre a performance, acessibilidade, boas práticas e SEO. A ferramenta pode ser utilizada tanto em aplicações para desktop como para dispositivos móveis, permitindo identificar áreas que necessitam de melhorias.

No âmbito do projeto **CultivUA**, utilizámos o Lighthouse para avaliar a qualidade da plataforma web, realizando testes específicos para as versões desktop e mobile. Os resultados obtidos atualmente são os seguintes:

Desktop:



Figura 37 - Resultado do teste de qualidade do lighthouse no desktop

Mobile:

Figura 38 - Resultado do teste de qualidade do lighthouse em mobile

Os resultados refletem o estado atual da aplicação, sendo este um projeto ainda em desenvolvimento. Identificámos que o principal ponto a ser melhorado é a performance, que recebeu uma pontuação mais baixa em ambas as plataformas. A acessibilidade apresenta uma classificação elevada, indicando que os utilizadores conseguem interagir com a aplicação de forma eficiente e inclusiva. As boas práticas e o SEO mostram resultados satisfatórios, mas ainda existem oportunidades para otimizar estes parâmetros.

No futuro, o objetivo da equipa será continuar a melhorar a plataforma, com especial atenção à performance, de forma a garantir tempos de carregamento mais rápidos e uma experiência de utilização mais fluida. Paralelamente, serão implementadas melhorias nos restantes parâmetros para maximizar a qualidade geral da aplicação e proporcionar aos utilizadores um serviço robusto, eficiente e de elevada qualidade. O uso contínuo do Lighthouse permitirá monitorizar estas evoluções, ajudando a medir o impacto das alterações realizadas e a guiar os próximos passos no desenvolvimento.

Os relatórios completos dos testes realizados com o Lighthouse estão disponíveis no **Anexo D** para consulta, permitindo verificar os detalhes das auditorias realizadas e orientar futuras melhorias no projeto.

19.4. Testes de usabilidade

Os testes de usabilidade são uma etapa crucial no desenvolvimento de qualquer aplicação, pois permitem avaliar como os utilizadores interagem com o sistema, identificar dificuldades, e recolher *feedback* para melhorar a experiência geral. Estes testes envolvem a observação de utilizadores reais enquanto realizam tarefas específicas, o que fornece dados importantes sobre a eficiência, eficácia e satisfação do utilizador ao utilizar a aplicação.

No âmbito do projeto **CultivUA**, foram realizadas duas sessões de testes de usabilidade em momentos distintos do desenvolvimento. A primeira situação utilizou o protótipo de alta-fidelidade da aplicação. Este protótipo, desenvolvido para representar o design final da plataforma, permitiu uma validação inicial da navegação e do layout, bem como a identificação de potenciais problemas antes do início da implementação da plataforma.

A segunda situação ocorreu já com a plataforma em desenvolvimento. Neste caso, os testes foram realizados com a aplicação funcional, permitindo observar como os utilizadores interagiam com as funcionalidades implementadas e recolher feedback sobre a experiência real de utilização.

Ambas as sessões de testes foram realizadas na unidade curricular de Desenvolvimento Web Multiplataforma, com membros da turma a participar como utilizadores. Durante os testes, os participantes seguiram um conjunto de tarefas escolhidas pelos membros do grupo do projeto. Estas tarefas tinham como objetivo reproduzir cenários de utilização real, cobrindo funcionalidades essenciais da plataforma, como o processo de login, a navegação entre páginas, a consulta de dados na dashboard e a simulação de interações com sensores IoT.

Os resultados dos testes forneceram *insights* valiosos para a equipa, permitindo ajustes no design, melhorias na interface do utilizador e a resolução de problemas de funcionalidade. Assim, os testes de usabilidade contribuíram significativamente para alinhar o desenvolvimento da plataforma com as necessidades e expectativas dos utilizadores finais.

19.4.1. Tabela de Tarefas Utilizadas nos Testes de Usabilidade

Com o objetivo de detalhar os cenários e ações que simulam o uso real da plataforma, foi criada uma tabela de tarefas para guiar os participantes em ambas as situações de teste: no protótipo de alta-fidelidade e na plataforma em desenvolvimento. Estas tarefas focaram-se em cenários quotidianos de interação com a aplicação, avaliando desde o registo até funcionalidades mais avançadas, como a interação com a loja e os sensores IoT. A tabela a seguir resume as tarefas utilizadas nos testes:

Tarefa	Objetivo	Passos Esperados
1. Criar uma Conta no CultivUA	Validar o processo de registo	Preenchimento do formulário com dados pessoais e confirmação de registo
2. Navegar pela Loja e Adicionar um Produto	Testar navegação e usabilidade da loja	Seleção de uma categoria, escolha de um produto e adição ao carrinho
3. Realizar Compra de um Produto	Verificar o processo de checkout	Adicionar um produto ao carrinho, inserção dos dados de envio e confirmação da compra
4. Encontrar Planta pelo Quiz	Testar o quiz de identificação de plantas	Responder às perguntas do quiz e visualizar o resultado da identificação
5. Identificar Planta por Fotografia	Testar o reconhecimento de plantas	O sistema processa a imagem e identifica a planta com base no reconhecimento de imagem.
6. Adicionar uma planta à dashboard	Validar adição da planta e da monitorização	Adicionar uma planta à conta e visualizar o processo

20. Limitações e Problemas Encontrados

Durante os testes de usabilidade realizados em aula, os testes de qualidade feitos anteriormente e a apresentação da aplicação ao professor da unidade curricular de Web Design, foram identificados diversos problemas específicos que impactaram diferentes áreas do projeto. Estes problemas incluem aspetos como a navegação, o design visual, a acessibilidade, a funcionalidade e a interação dos utilizadores com a aplicação. Para além disso, foram também evidenciadas algumas limitações no próprio projeto, relacionadas com decisões iniciais de design ou restrições técnicas que comprometeram a experiência do utilizador e a eficácia da aplicação. Esta análise detalhada dos problemas e limitações visa fornecer uma compreensão clara dos desafios encontrados e orientar melhorias para versões futuras.

20.1. Problemas Encontrados nos Testes de Qualidade e Ações de Melhoria

Com base na avaliação realizada através do Google Lighthouse e nos resultados apresentados, existem várias áreas de melhoria que podem ser abordadas para otimizar a qualidade geral da plataforma **CultivUA**. Embora todos os parâmetros (Performance, Acessibilidade, Boas Práticas e SEO) possam ser aprimorados, a performance, que obteve a pontuação mais baixa, deve ser o foco principal para garantir uma experiência mais fluida e eficiente para os utilizadores.

Os problemas identificados na avaliação da performance indicam que a plataforma pode ser significativamente melhorada com intervenções estratégicas. A presença de JavaScript e CSS não utilizados contribui para o aumento do tempo de carregamento. A aplicação de técnicas como *tree shaking* e *purge* CSS pode ajudar a eliminar código desnecessário, reduzindo o tamanho dos ficheiros carregados. Além disso, a otimização de recursos, como JavaScript e CSS. Este processo remove espaços em branco, comentários e outros elementos redundantes nos ficheiros.

Outra área crítica é a otimização de imagens. Utilizar formatos de imagem de próxima geração, como WebP, e redimensionar imagens conforme necessário. Além disso, assegurar que todas as imagens têm largura e altura explícitas e ajuda a evitar deslocamentos de layout. A eliminação de recursos que bloqueiam a renderização, como ficheiros JavaScript e CSS,

também deve ser priorizada. Esses recursos podem ser adiados ou carregados de forma assíncrona, utilizando ferramentas como o Webpack para implementar essas práticas.

A configuração de políticas eficientes de cache para recursos estáticos é essencial para evitar recarregamentos desnecessários e melhorar a reutilização de ficheiros. Além disso, a compressão de texto com Gzip ou Brotli e a codificação eficiente de imagens podem reduzir significativamente o tamanho dos dados transferidos pela rede. É essencial rever os recursos carregados, removendo aqueles que não são cruciais ou carregando-os condicionalmente conforme o contexto do utilizador.

Embora a performance seja o foco principal, outras áreas identificadas no relatório também requerem atenção. Na acessibilidade, é necessário melhorar a semântica da aplicação ao adicionar nomes acessíveis aos botões, corrigir problemas de contraste entre texto e fundo e reorganizar os elementos de cabeçalho para seguir uma hierarquia lógica, beneficiando utilizadores com deficiências e melhorando a usabilidade global.

Nas boas práticas, resolver erros do navegador e reduzir a dependência de cookies e bibliotecas de terceiros contribuirá para uma aplicação mais segura e eficiente. No SEO, garantir que o documento contém uma meta descrição adequada e que os links são rastreáveis por motores de busca aumentará a visibilidade da aplicação nos resultados de pesquisa.

É importante destacar que a plataforma está em fase de desenvolvimento contínuo, e estas avaliações refletem o estado atual da aplicação. O objetivo para o futuro é implementar as melhorias necessárias, com especial atenção à performance, uma vez que afeta diretamente a experiência do utilizador. Além disso, intervenções sistemáticas em acessibilidade, boas práticas e SEO irão consolidar a qualidade da plataforma, proporcionando uma solução robusta e eficiente para os utilizadores. Por fim, todos os testes realizados com o Lighthouse estão disponíveis no **Anexo D**, permitindo uma análise detalhada dos problemas detetados e das auditorias realizadas. Isso assegura transparência no processo e facilita a monitorização dos progressos alcançados ao longo do desenvolvimento.

20.2. Resultados dos Testes de Usabilidade e Ações de Melhoria

Os testes de usabilidade destacaram tanto pontos fortes quanto áreas de melhoria na experiência oferecida pelo CultivUA. Este feedback foi essencial para orientar ajustes e evoluções, focados em tornar a plataforma mais intuitiva, acessível e funcional. Seguem-se os problemas identificados e as medidas propostas para a sua resolução, de forma a satisfazer os utilizadores finais.

20.2.1. Problemas Identificados

1. Criação de Conta e Login

- **Problema:** Dúvidas sobre o comprimento mínimo e máximo da palavra-passe durante o registo.
- **Impacto:** Pode causar frustração nos utilizadores e dificultar o processo de criação de conta.
- **Medida:** Incluir instruções claras sobre os requisitos da palavra-passe abaixo do *input* (por exemplo, "A palavra-passe deve ter entre 8 e 16 caracteres, incluindo pelo menos uma letra maiúscula e um número") diretamente no formulário de registo.

2. Layout da Landing Page

- **Problema:** Os utilizadores não conseguiriam perceber facilmente que há mais conteúdo abaixo da dobra inicial da página.
- **Impacto:** Reduz a taxa de exploração do website e afeta negativamente a experiência do utilizador.
- **Medida:** Adicionar um ícone de seta para baixo ou outra indicação visual que sugira a existência de mais conteúdo. Além disso, pode-se ajustar o espaçamento e o design para "convidar" o *scroll*.

3. Exibição da Opção "Dashboard"

- **Problema:** A opção "Dashboard" aparece para utilizadores que não iniciaram sessão.
- **Impacto:** Pode causar confusão e criar uma expectativa de acesso não autorizado.
- **Medida:** Ocultar o menu "Dashboard" quando a sessão não estiver iniciada, usar uma validação no *frontend* para verificar o estado de autenticação.

4. Falta do Botão de Logout na Dashboard

- **Problema:** Ausência de uma opção clara para terminar a sessão na dashboard.
- **Impacto:** Dificulta o encerramento da sessão e compromete a usabilidade.
- **Medida:** Adicionar um botão de *logout* visível na dashboard.

5. Sugestões dos Utilizadores

- **Problema:** Necessidade de melhorar a experiência geral com a integração de funcionalidades adicionais, como:
 - Apresentar o quiz no menu lateral da dashboard.
 - Visualização de mais opções nos resultados do quiz.
 - *Chatbot* no menu lateral da dashboard.
- **Medida:** Priorizar o desenvolvimento destas funcionalidades em futuras iterações, alinhando-as com os objetivos do projeto e o feedback dos utilizadores.

20.2.2. Ações de Melhoria

Com base nos problemas identificados, foram delineadas ações de melhoria para culmar cada ponto identificado:

1. **Formulários:** Atualizar as instruções e as validações no *frontend* e *backend* para tornar o processo de registo mais claro e acessível.
2. **Landing Page:** Implementar elementos visuais que promovam a exploração da página, como animações suaves ou ícones indicativos.
3. **Dashboard:** Ajustar as condições de exibição do menu e adicionar as funcionalidades em falta para melhorar a experiência de navegação.
4. **Funcionalidades Futuras:** Planear o desenvolvimento e implementação das funcionalidades adicionais sugeridas pelos utilizadores, de maneira a garantir uma evolução contínua e alinhada com as necessidades reais dos mesmos.

A identificação de problemas através dos testes de usabilidade foi fundamental para ajustar e melhorar o **CultivUA**. Ao implementar as medidas propostas, espera-se não apenas corrigir as falhas identificadas, mas também oferecer uma experiência mais completa e satisfatória para os utilizadores, consolidando os objetivos do projeto.

20.3. Problemas Identificados Através do Feedback do Professor

Aqui são apresentados os problemas identificados durante os testes de usabilidade e a avaliação do projeto pelo professor da unidade curricular de Web Design. Estes problemas variam desde dificuldades na interação e navegação até inconsistências no design e limitações técnicas. A identificação destes pontos críticos é essencial para compreender as áreas que necessitam de melhorias, garantindo uma aplicação mais funcional, intuitiva e alinhada com os objetivos propostos.

Problema de legibilidade causado pelo uso de verde-claro nos textos

Com base no feedback recebido, foi identificado um problema relacionado com a legibilidade em algumas partes do projeto onde foram utilizados tons de verde-claro para os textos, especialmente em elementos mais pequenos ou com menor contraste. Apesar de a escolha da cor estar alinhada com a paleta proposta, esta apresenta dificuldades práticas para os utilizadores em situações específicas, como a leitura em dispositivos móveis ou em ambientes com luz solar direta.

A baixa legibilidade causada pela cor afeta diretamente a acessibilidade e a usabilidade do mesmo. Textos mais claros em fundos com pouco contraste podem exigir maior esforço visual, tornando a experiência de leitura mais cansativa e, em alguns casos, inviabilizando a compreensão das informações. Este problema é ainda mais significativo para utilizadores com dificuldades visuais, como daltonismo ou baixa visão, que podem enfrentar barreiras adicionais ao aceder ao conteúdo.

Para melhorar a acessibilidade e a experiência do utilizador, é essencial rever as combinações de cores utilizadas nos textos, particularmente nos elementos mais pequenos ou críticos para a navegação. Substituir o verde-claro por tons mais escuros ou aumentar o contraste poderá garantir que todos os utilizadores consigam aceder ao conteúdo do relatório de forma eficiente, independentemente do dispositivo utilizado ou das condições de iluminação.

Informações desnecessárias nas tabelas da Dashboard admin

Com o feedback obtido, foi identificado um problema relacionado com a presença de informações desnecessárias nas tabelas presentes na Dashboard admin, como campos de ID, *threshold* e outros dados técnicos que não eram relevantes para os utilizadores. A inclusão desses elementos acabou por criar uma sobrecarga de informação e de conteúdo na própria interface do utilizador, tornando esta mais complexa e por consequência mais difícil de compreender.

Este excesso de informações afeta diretamente a usabilidade, uma vez que os administradores precisam de mais tempo para localizar as informações essenciais, o que aumenta a dificuldade em realizar tarefas simples e reduz a eficiência do sistema. A exibição de dados irrelevantes nas tabelas cria confusão e dificulta a análise rápida das informações que realmente importam para os administradores.

No que diz respeito à acessibilidade, o problema torna-se ainda mais relevante. Para administradores com limitações ou dificuldades de navegação, o excesso de dados técnicos e irrelevantes pode resultar numa maior dificuldade em identificar o que é realmente necessário. Isso pode comprometer a experiência de navegação, tornando o sistema mais difícil de usar e prejudicando a eficácia da aplicação. Para melhorar, seria importante simplificar as tabelas e remover as informações que não são pertinentes para os administradores, tornando a aplicação mais intuitiva e acessível para uma melhor gestão.

Problema de ausência de simbologia no preço

No âmbito do projeto, que foi desenvolvido em língua portuguesa, que será alterada para inglesa, pois o projeto poderá alcançar um público internacional, foi identificado um problema relacionado com a ausência de simbologia nos preços apresentados. Este detalhe, embora aparentemente menor, torna-se significativo ao considerar que o público-alvo pode incluir utilizadores de países onde o euro não é a moeda utilizada.

A falta da indicação explícita de que os preços estão em euros pode causar confusão entre os utilizadores internacionais. Estes podem não compreender imediatamente a moeda em que os valores estão expressos, levando a dúvidas e dificultando a tomada de decisão durante a

navegação ou a compra. Este problema pode afetar a experiência do utilizador e, consequentemente, a perceção do projeto no mercado global.

Para resolver esta questão e melhorar a clareza e acessibilidade da plataforma, seria importante incluir sempre a simbologia monetária (€ ou outro) nos preços apresentados. Esta simples adição ajudará a eliminar ambiguidades e a garantir que o projeto é mais intuitivo e transparente para todos os utilizadores, independentemente do seu país de origem ou da moeda com que estão habituados a trabalhar.

20.4. Limitações no projeto

Embora o projeto **CultivUA** tenha alcançado progressos significativos no desenvolvimento de uma plataforma inovadora para a promoção da agricultura urbana, é importante reconhecer as limitações que surgiram ao longo do seu ciclo de desenvolvimento. Estas limitações resultam de vários fatores, incluindo o tempo disponível, os recursos técnicos, as ferramentas utilizadas e a própria complexidade dos objetivos propostos.

Este tópico tem como objetivo identificar e discutir as principais restrições enfrentadas, analisando o impacto que tiveram no desenvolvimento do projeto. Reconhecer estas limitações é fundamental para avaliar o estado atual da plataforma e traçar estratégias para melhorias futuras, garantindo que o **CultivUA** continue a evoluir como uma solução robusta e eficiente.

Problema de Limitação na API de Identificação de Plantas

Durante a análise das funcionalidades da plataforma, foi identificado um problema relacionado com a limitação da versão gratuita de uma API utilizada (Perenual API) para a identificação de plantas. Atualmente, esta API apenas permite a consulta de plantas cujos identificadores (IDs) estão entre os primeiros 3000 registos. Caso o ID de uma planta esteja fora deste intervalo, ou seja, acima de 3000, a funcionalidade deixa de estar disponível na versão gratuita.

Esta restrição apresenta-se como um entrave significativo, uma vez que compromete a experiência dos utilizadores que procuram informações sobre as plantas. Esta limitação pode gerar frustração, já que os utilizadores podem criar a expectativa de que a plataforma

disponibiliza um serviço completo, mas deparam-se com barreiras técnicas que impedem o acesso a uma parte substancial dos dados.

A resolução deste problema permitirá não apenas melhorar a fiabilidade e a consistência da plataforma, como também que os utilizadores conseguem obter as informações que procuram, independentemente do ID da planta em questão. Este passo será fundamental para assegurar o sucesso da plataforma no mercado global.

21. Análise de resultados

Ao analisar a lista de requisitos previamente mencionada verificou-se que foram levantados 66 requisitos no total, sem distinção entre funcionais e não funcionais.

Como se observa no gráfico da [Figura 39 - Gráfico de Requisitos Cumpridos | Não Cumpridos | Parcialmente Cumpridos] em termos quantitativos pode-se afirmar que foram cumpridos 50 ($\approx 76\%$) dos requisitos iniciais, tendo 2 ($\approx 3\%$) ficado parcialmente cumpridos e 14 ($\approx 21\%$) por implementar.

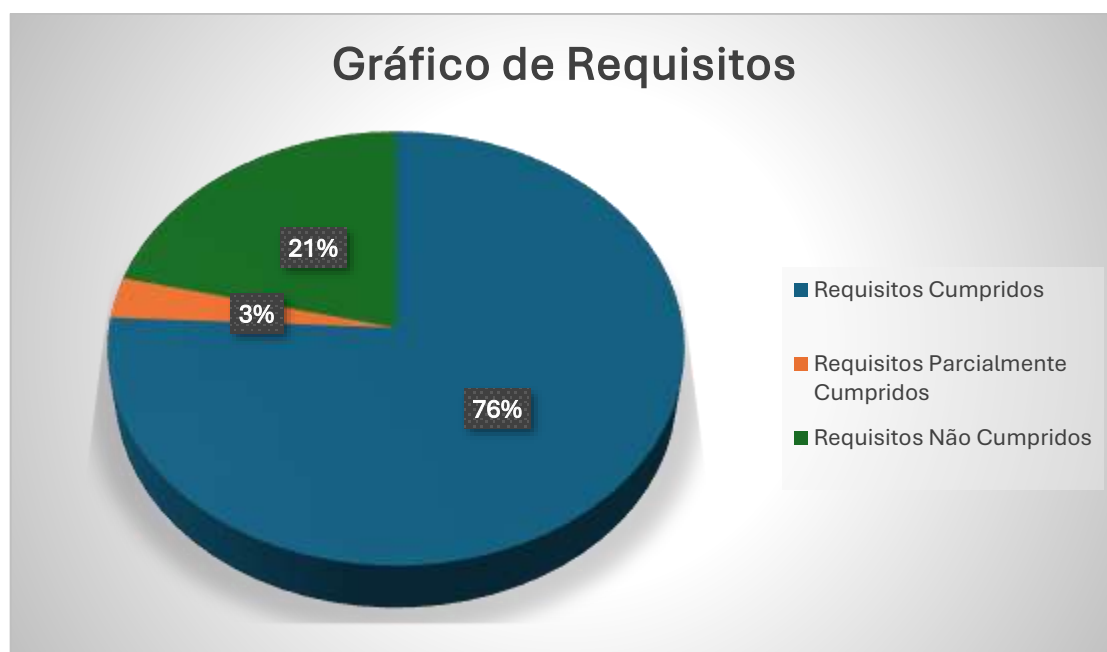


Figura 39 - Gráfico de Requisitos Cumpridos | Não Cumpridos | Parcialmente Cumpridos

Requisitos não cumpridos

RF.10: O sistema confirma a compra e envia notificações por email com o resumo da compra.

RF.16: O utilizador pode registar-se utilizando redes sociais (opcional).

RF.18: O utilizador pode recuperar a palavra-passe através de email.

RF.30: O utilizador pode visualizar dados históricos e tendências de crescimento através de gráficos.

RF.32: O utilizador pode configurar as preferências de alertas (frequência, canal de notificação).

RF.33: O sistema gera relatórios de desempenho do crescimento das plantas.

RF.35: O sistema apresenta comparações de crescimento das plantas ao longo do tempo.

RF.39: O administrador pode configurar os intervalos de recolha de dados dos sensores.

RF.45: O administrador pode gerir as páginas de FAQ, política de privacidade e termos de serviço.

RF.46: O utilizador pode gerar relatórios personalizados sobre o estado das suas plantas (baseados em dados históricos de crescimento e condições ambientais).

RF.47: O sistema permite que o utilizador faça download dos relatórios em formatos PDF.

RF.49: O sistema gera relatórios de desempenho da plataforma, incluindo tempo de resposta dos sensores e latência de dados.

RNF.7: Os dados dos sensores devem ser encriptados ao serem enviados para o servidor.

RNF.8: A plataforma deve implementar autenticação de dois fatores (2FA) para contas de utilizadores e administradores.

22. Reflexão Crítica e Conclusões

Este tópico tem como objetivo analisar o percurso do projeto **CultivUA**, destacando os principais aprendizados, desafios e conquistas. Este segmento foca-se em dois subtemas principais: "Sugestões para o Futuro", onde são apresentadas ideias e oportunidades para o desenvolvimento e aperfeiçoamento da plataforma, e "Síntese das Experiências", que reflete sobre as vivências do grupo durante a realização do projeto.

Esta abordagem permite não apenas avaliar o impacto do trabalho realizado, mas também identificar caminhos para a evolução do projeto e consolidar as aprendizagens adquiridas ao longo deste processo enriquecedor.

22.1. Sugestões para o futuro

À medida que o **CultivUA** evolui, identificaram-se diversas oportunidades para expandir e melhorar a plataforma. Estas ideias futuras focam-se em proporcionar uma experiência mais integrada, eficiente e adaptada às necessidades dos utilizadores, bem como em corrigir aspetos que ficaram por implementar na versão atual.

Expansão de Funcionalidades

- **Sistema de Irrigação Automatizada**

A integração de um sistema de irrigação automática permitiria otimizar o cuidado das plantas. Este sistema utilizaria dados recolhidos pelos sensores de humidade e temperatura para determinar automaticamente o momento e a quantidade de água necessária para cada planta. Tal abordagem reduziria desperdícios e promoveria práticas sustentáveis de cultivo.

- **Suporte Modular para Sensores**

Pretendemos desenvolver uma arquitetura modular que suporte diferentes tipos de sensores, como sensores de pH e condutividade elétrica. Esta modularidade permitiria aos utilizadores configurar os seus kits de monitorização de acordo com as necessidades específicas de cada planta, ampliando as funcionalidades da plataforma.

- **Automação de Processos Avançados**

A integração com dispositivos IoT adicionais, como válvulas inteligentes e sistemas de iluminação, possibilitará a automação de processos de cultivo mais complexos, criando um ambiente controlado ideal para o crescimento das plantas.

- **Vaso Inteligente**

O desenvolvimento de um vaso inteligente, produto exclusivo do **CultivUA**, incluirá:

- Um espaço dedicado para integrar sensores modulares que monitorizem condições como humidade, temperatura e luminosidade.
- Um LED inteligente que muda de cor para indicar as necessidades da planta (como falta de água ou excesso de luz).
- Um design otimizado para facilitar o uso com a plataforma, permitindo que os dados sejam sincronizados automaticamente.

- **Correção e Implementação de Funcionalidades Inacabadas**

Serão priorizadas as funcionalidades identificadas como inacabadas na versão atual, garantindo que a plataforma ofereça uma experiência completa e sem falhas.

Sustentabilidade e Integração Comunitária

O futuro do **CultivUA** também passará pela promoção de uma agricultura urbana mais inclusiva e sustentável, com funcionalidades que incentivem a partilha de recursos e conhecimentos entre utilizadores, como fóruns.

22.2. Síntese das Experiências

O Projeto Temático em Desenvolvimento Web destacou-se como uma oportunidade única para integrar e aplicar os conhecimentos adquiridos nas várias disciplinas que compõem o módulo temático. Este projeto proporcionou-nos uma experiência prática e colaborativa, permitindo explorar o desenvolvimento de um produto enquanto trabalhávamos em equipa para alcançar objetivos específicos.

A dinâmica do grupo foi marcada por um espírito de cooperação e ajuda mútua, mesmo nos momentos mais exigentes do projeto. A comunicação aberta entre os membros foi essencial para superar desafios e manter o foco na entrega de um trabalho de qualidade. Este ambiente colaborativo não só contribuiu para o sucesso do projeto, mas também reforçou competências como o trabalho em equipa, a resolução de problemas e a gestão de tarefas.

No geral, a experiência foi extremamente positiva, permitindo-nos aplicar conhecimentos técnicos e desenvolver habilidades interpessoais essenciais para projetos futuros, tanto no contexto académico quanto no profissional.

23. Conclusão

A realização do projeto **CultivUA** proporcionou uma oportunidade única para aplicar de forma prática os conhecimentos adquiridos ao longo do curso, especialmente nas unidades curriculares relacionadas com o Projeto Temático, Desenvolvimento Web Multiplataforma e Web Design, sob a orientação dos docentes Fábio Marques e José Martins. Estas disciplinas foram fundamentais não apenas para o desenvolvimento técnico, mas também para o planeamento inicial e a execução organizada do projeto.

Durante o processo, enfrentámos uma série de desafios que contribuíram para o nosso crescimento académico e profissional. De forma geral, tivemos de explorar novas ferramentas e tecnologias, sendo que, para muitos de nós, esta foi a primeira experiência com uma *framework* como o Angular. No início, sentimos dificuldades com esta tecnologia, dado que nunca tínhamos trabalhado com *frameworks* semelhantes. Contudo, com dedicação e trabalho em equipa, conseguimos superar estas barreiras e melhorar as nossas competências nesta área.

Outro aspeto desafiador foi a conexão dos sensores ao Arduino e a integração dos dados na base de dados. Este foi um dos pontos mais técnicos e inovadores do projeto, exigindo pesquisa, experimentação e resolução de problemas em tempo real. Através deste processo, conseguimos adquirir uma compreensão mais profunda sobre a comunicação entre dispositivos IoT e sistemas baseados em web.

Além disso, a necessidade de explorar e aprender sobre novas ferramentas e conceitos foi uma vantagem para o nosso futuro. A experiência acumulada neste projeto não só nos preparou para enfrentar desafios semelhantes no mercado de trabalho, como também reforçou a nossa capacidade de adaptação e aprendizagem contínua.

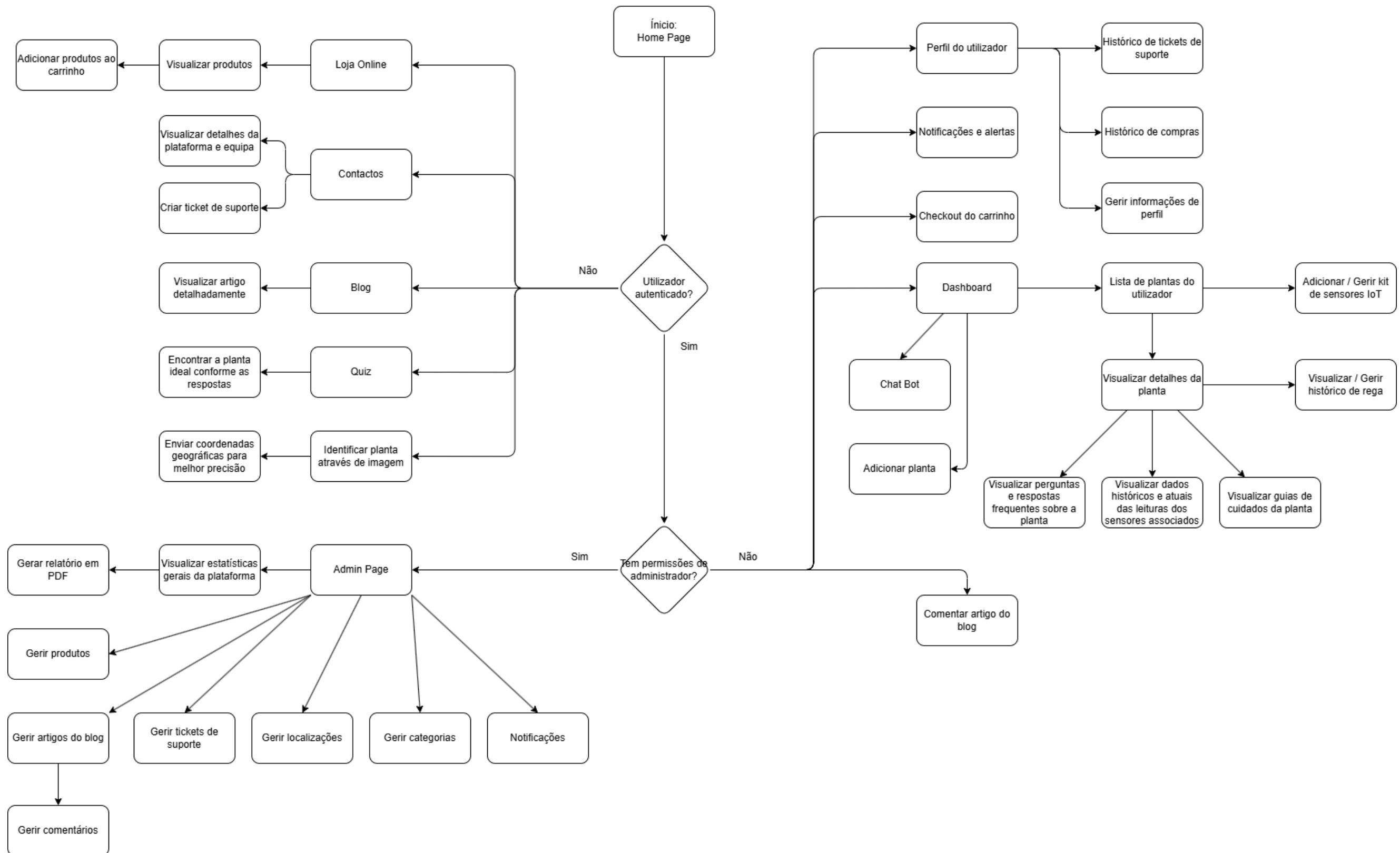
Em suma, o projeto **CultivUA** representou um excelente desafio, que testou e fortaleceu as nossas competências técnicas, criatividade e trabalho em equipa. Apesar das dificuldades encontradas, o resultado final foi extremamente gratificante, comprovando a importância de uma abordagem colaborativa e persistente. Este percurso deixou-nos mais preparados para futuros projetos e consolidou ainda mais a amizade e a ética de trabalho entre os membros do grupo.

24. Bibliografia

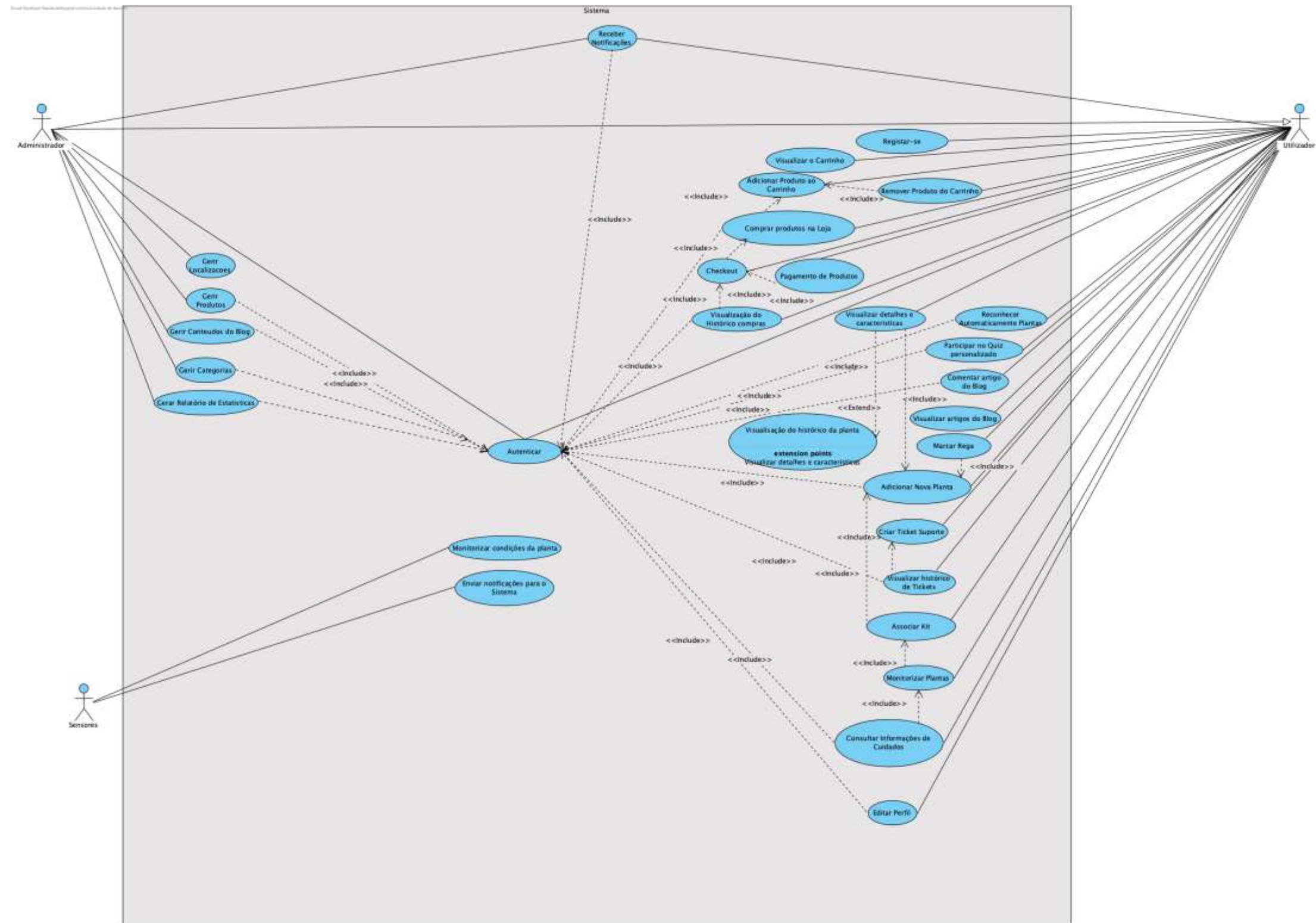
- [1] «What is Laravel? Explain it like I'm five», *RunCloud Blog*. Disponível em: <https://runcloud.io/blog/what-is-laravel>.
- [2] «Angular - What is Angular?», Disponível em: <https://v17.angular.io/guide/what-is-angular>.
- [3] «plant.id API v3», *Plant.id v3*. Disponível em: <https://documenter.getpostman.com/view/24599534/2s93z5A4v2>.
- [4] Taesang, «Free Plant API | Houseplants,Garden,Trees,Flowers,Images & Data». Disponível em: <https://perenual.com/docs/api>.
- [5] «Spike Angular Admin Template - WrapPixel». Disponível em: <https://www.wrappixel.com/templates/spike-angular-admin-template/>.
- [6] «Stripe API Reference». Disponível em: <https://docs.stripe.com/api>.
- [7] «Bump.sh API documentation». Disponível em: <https://developers.bump.sh/>.
- [8] «FarmBot | Open-Source CNC Farming», *FarmBot*. Disponível em: <https://farm.bot/>.
- [9] «Click & Grow», *Click & Grow EU*. Disponível em: <https://eu.clickandgrow.com/pages/indoor-gardens>.
- [10] «Smart System GARDENA». Disponível em: <https://www.gardena.com/pt/produtos/smart>.
- [11] «Docker», *Docker Documentation*, 50:11 - -0800 de 800. Disponível em: <https://docs.docker.com/>.
- [12] «MySQL :: MySQL Documentation». Disponível em: <https://dev.mysql.com/doc/>.
- [13] N. Ahmed, «Information Architecture:The Comprehensive Guide to Information Architecture», *Medium*, 22 de setembro de 2023. Disponível em: <https://medium.com/@nasir-ahmed03/information-architecture-the-comprehensive-guide-to-information-architecture-3912cc671296>.
- [14] F. Hanif, «Understanding the MVC Architecture in Laravel: A Comprehensive Guide», *Medium*, 26 de junho de 2023. Disponível em: <https://fkrihnif.medium.com/understanding-the-mvc-architecture-in-laravel-a-comprehensive-guide-8f620cc139b6>.
- [15] «Jasmine Documentation». Disponível em: <https://jasmine.github.io/>. [Acedido: 2 de janeiro de 2025]
- [16] «Karma - Spectacular Test Runner for Javascript». Disponível em: <https://karma-runner.github.io/latest/index.html>.
- [17] N. Maduro, «Installation | Pest - The elegant PHP Testing Framework». Disponível em: <http://pestphp.com/docs/installation>.
- [18] «Introduction to Lighthouse», *Chrome for Developers*. Disponível em: <https://developer.chrome.com/docs/lighthouse/overview>.

Anexos

Anexo A



Anexo B



Anexo C

Loja Online

1. Adicionar Produto ao Carrinho

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador seleciona um produto e adiciona-o ao carrinho de compras, podendo ajustar a quantidade conforme necessário.
- **Pré-condições:** O utilizador deve estar autenticado.
- **Fluxo Principal:**
 1. O utilizador navega pela loja e escolhe um produto.
 2. O utilizador escolhe a quantidade desejada.
 3. O utilizador clica em "Adicionar ao Carrinho".
 4. O sistema adiciona o produto ao carrinho e atualiza a quantidade.
- **Fluxo Alternativo:** O produto não está disponível em stock. O sistema exibe uma mensagem de erro.
- **Pós-condições:** O produto é adicionado ao carrinho do utilizador.

2. Remover Produto do Carrinho

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador remove itens do carrinho antes de finalizar a compra.
- **Pré-condições:** O utilizador tem produtos no carrinho.
- **Fluxo Principal:**
 1. O utilizador visualiza o carrinho.

2. O utilizador seleciona o produto a remover.
 3. O utilizador clica em "Remover".
 4. O sistema remove o produto do carrinho.
- **Fluxo Alternativo:** O carrinho está vazio. O sistema exibe uma mensagem informando que não há produtos.
 - **Pós-condições:** O produto é removido do carrinho.

3. Visualizar o Carrinho

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador visualiza o resumo dos produtos no carrinho, incluindo preços, quantidades e totais.
- **Pré-condições:** O utilizador tem produtos no carrinho.
- **Fluxo Principal:**
 1. O utilizador clica no ícone do carrinho.
 2. O sistema exibe a página com o resumo dos produtos, incluindo preço e quantidade.
- **Pós-condições:** O utilizador vê todos os detalhes dos produtos no carrinho.

4. Checkout

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador conclui a compra inserindo as suas informações pessoais e de pagamento.
- **Pré-condições:** O utilizador tem produtos no carrinho.
- **Fluxo Principal:**
 1. O utilizador clica em "Finalizar Compra".

2. O utilizador insere os dados de envio e pagamento.
 3. O sistema processa o pagamento e exibe uma confirmação.
 4. O utilizador recebe um email de confirmação.
- **Fluxo Alternativo:** O pagamento falha. O sistema exibe uma mensagem de erro e pede para tentar novamente.
 - **Pós-condições:** A compra é finalizada e os produtos serão enviados.

5. Pagamento de Produtos

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador paga pelos produtos utilizando métodos como cartão de crédito ou transferência bancária.
- **Pré-condições:** O utilizador tem produtos no carrinho e inseriu informações de pagamento.
- **Fluxo Principal:**
 1. O utilizador escolhe o método de pagamento.
 2. O utilizador insere os detalhes do pagamento.
 3. O sistema processa o pagamento.
 4. O utilizador recebe uma notificação de pagamento bem-sucedido.
- **Fluxo Alternativo:** O pagamento é recusado. O sistema solicita correção dos dados de pagamento.
- **Pós-condições:** O pagamento é concluído.

6. Comprar Produtos na Loja

- **Ator Principal:** Utilizador

- **Descrição:** O utilizador realiza a compra de produtos na loja online, seguindo o processo de checkout.
- **Pré-condições:** O utilizador deve ter produtos no carrinho e estar autenticado.
- **Fluxo Principal:**
 1. O utilizador seleciona os produtos no carrinho e clica em "Finalizar Compra".
 2. O utilizador insere as informações de envio e pagamento.
 3. O sistema processa o pagamento e confirma a compra.
 4. O utilizador recebe uma confirmação por email.
- **Fluxo Alternativo:** O pagamento falha. O sistema exibe uma mensagem de erro e solicita novos dados de pagamento.
- **Pós-condições:** A compra é concluída, e os produtos são enviados ao utilizador.

Gestão de Utilizadores

7. Registar-se

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador cria uma conta ao fornecer um email e uma palavra-passe.
- **Pré-condições:** O utilizador não tem uma conta existente.
- **Fluxo Principal:**
 1. O utilizador clica em "Registar-se".
 2. O utilizador preenche o formulário com email e palavra-passe.
 3. O sistema cria a conta e envia um email de confirmação.
- **Fluxo Alternativo:** O email já está em utilização. O sistema exibe uma mensagem de erro.
- **Pós-condições:** O utilizador tem uma conta criada.

8. Autenticar

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador entra na plataforma com email e palavra-passe.
- **Pré-condições:** O utilizador tem uma conta.
- **Fluxo Principal:**
 1. O utilizador insere as credenciais.
 2. O sistema valida as credenciais e autentica o utilizador.
- **Fluxo Alternativo:** As credenciais são inválidas. O sistema exibe uma mensagem de erro.
- **Pós-condições:** O utilizador está autenticado.

9. Editar Perfil

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador pode atualizar as suas informações pessoais, como nome e email.
- **Pré-condições:** O utilizador está autenticado.
- **Fluxo Principal:**
 1. O utilizador acede ao perfil.
 2. O utilizador atualiza as informações.
 3. O sistema guarda as alterações.
- **Pós-condições:** O perfil do utilizador é atualizado.

10. Consultar Histórico de Compras

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador pode visualizar o histórico de todas as compras realizadas na plataforma.
- **Pré-condições:** O utilizador deve estar autenticado e ter compras realizadas na conta.
- **Fluxo Principal:**
 1. O utilizador acede ao seu perfil e seleciona a opção "Histórico de Compras".
 2. O sistema apresenta uma lista de todas as compras efetuadas, com informações detalhadas sobre os produtos e datas.
- **Fluxo Alternativo:** O utilizador não tem compras anteriores. O sistema exibe uma mensagem informando que não há histórico de compras.
- **Pós-condições:** O utilizador consulta o histórico de compras efetuadas.

Monitorização e Gestão de Plantas

11. Adicionar Nova Planta

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador pode adicionar novas plantas ao seu perfil.
- **Pré-condições:** O utilizador está autenticado.
- **Fluxo Principal:**
 1. O utilizador insere o código da planta.
 2. O sistema valida o código e adiciona a planta ao perfil do utilizador.
- **Pós-condições:** A planta é adicionada ao perfil do utilizador.

12. Consultar Informações de Cuidados

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador pode adicionar ou consultar informações manuais sobre os cuidados com a planta, como rega, fertilização e poda.
- **Pré-condições:** O utilizador tem plantas registadas no sistema.
- **Fluxo Principal:**
 1. O utilizador acede ao perfil de uma planta.
 2. O utilizador consulta as informações sobre os cuidados recomendados ou insere informações personalizadas.
 3. O sistema guarda as novas informações.
- **Pós-condições:** As informações sobre cuidados são atualizadas ou consultadas pelo utilizador.

13. Visualizar Detalhes e Características da Planta

Ator Principal: Utilizador

Descrição:

O utilizador pode aceder às informações detalhadas sobre uma planta específica, incluindo condições ideais de cultivo, dimensões, características adicionais (como toxicidade, manutenção necessária, ciclo de vida, presença de frutas ou flores), usos (medicinais, comestíveis, etc.) e dicas de utilização.

Pré-condições:

- A planta está registada na base de dados da plataforma.
- O utilizador tem acesso à interface de consulta de plantas com a autenticação feita.

Fluxo Principal:

1. O utilizador pesquisa ou seleciona uma planta a partir da lista disponível na *dashboard*.
2. O sistema exibe os seguintes detalhes da planta:
 - **Condições ideais:** luz, temperatura, umidade, tipo de solo, frequência de rega.

- **Dimensões:** altura e largura típicas da planta quando adulta.
- **Características adicionais:**
 - Necessidade de manutenção (Ex.: alta, média, baixa).
 - Toxicidade (Ex.: tóxica ou segura para humanos e animais).
 - Presença de frutos ou flores e descrição de suas características.
 - Ciclo de vida (Ex.: anual, perene).
- **Usos da planta:**
 - Medicinal, comestível etc.
- **Dicas de utilização:**
 - Sugestões práticas.

Pós-condições:

- O utilizador obteve todas as informações desejadas sobre a planta selecionada.

14. Visualização do Histórico de Plantas --

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador pode visualizar o histórico completo de crescimento de cada planta com base em dados recolhidos pelos sensores e registos manuais.
- **Pré-condições:** O utilizador tem plantas registadas e sensores instalados.
- **Fluxo Principal:**
 1. O utilizador acede ao histórico de uma planta específica.
 2. O sistema apresenta gráficos e dados detalhados sobre as condições da planta.
- **Pós-condições:** O utilizador visualiza o histórico da condição da planta.

15. Regar Plantas com Base em Calendário

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador regista no calendário a data em que regou a planta, e o sistema calcula e avisa quando será necessário regar novamente.
- **Pré-condições:** O utilizador tem plantas registadas e sensores instalados.
- **Fluxo Principal:**

1. O utilizador tem plantas registadas e configurou o ciclo de rega de cada planta (frequência recomendada).
 2. O sistema tem acesso ao histórico de rega da planta.
- **Pós-condições:** O sistema mantém um registo atualizado e o utilizador visualiza a próxima data de rega no calendário.

16. Monitorizar Plantas

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador pode monitorizar as condições das suas plantas (temperatura, humidade, luminosidade) através de gráficos na *dashboard*, com dados fornecidos pelos sensores conectados.
- **Pré-condições:** O utilizador tem plantas e sensores registados.
- **Fluxo Principal:**
 1. O utilizador acede à secção de monitorização de plantas.
 2. O sistema apresenta gráficos e dados atualizados dos sensores conectados.
- **Pós-condições:** O utilizador visualiza os dados das plantas em tempo real.

17. Receber Notificações e Alertas --

- **Ator Principal:** Utilizador
- **Descrição:** O sistema envia notificações ao utilizador quando as plantas precisam de água, luz ou se os parâmetros ambientais estão fora dos níveis ideais.
- **Pré-condições:** O utilizador tem plantas registadas e sensores instalados.
- **Fluxo Principal:**
 1. O sistema monitoriza os sensores conectados.
 2. O sistema identifica se há condições fora dos parâmetros.

3. O sistema envia uma notificação ao utilizador.

- **Fluxo Alternativo:** O utilizador não ativou notificações. O sistema não envia alertas.
- **Pós-condições:** O utilizador é notificado sobre o estado das plantas.

18. Reconhecer Automaticamente Plantas

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador pode identificar plantas através de uma foto, com a ajuda de uma API externa de reconhecimento.
- **Pré-condições:** O utilizador tem uma foto da planta.
- **Fluxo Principal:**
 1. O utilizador faz upload da foto da planta.
 2. O sistema envia a imagem para uma API de reconhecimento.
 3. A API devolve informações sobre a planta.
 4. O utilizador recebe os resultados na plataforma.
- **Pós-condições:** A planta é identificada e registada no perfil do utilizador.

19. Consultar informações e cuidados

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador pode consultar informações e cuidados da sua planta.
- **Pré-condições:** O utilizador tem de ter uma planta.
- **Fluxo Principal:**
 1. O utilizador seleciona ver mais na sua planta.
 2. O utilizador pode consultar as informações sobre as plantas e os seus cuidados.

20. Associar kit

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador adiciona um kit à sua planta para monitorizar dados sobre a mesma.
- **Pré-condições:** O utilizador tem de ter uma planta e o kit de sensores conectados corretamente.
- **Fluxo Principal:**
 1. O utilizador seleciona adicionar kit na sua planta.
 2. O utilizador insere o número do kit e o nome.
 3. O utilizador agora recebe informações de temperatura humidade etc.
- **Pós-condições:** O kit tem de continuar ligado a planta de modo a obter dados verdadeiros e corretos.

21. Criar ticket suporte

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador cria um ticket de suporte para contactar a equipa.
- **Pré-condições:** O utilizador tem de estar com o Log in feito para poder enviar um Ticket.
- **Fluxo Principal:**
 1. O utilizador preenche todos os campos necessários.
 2. O utilizador enviar o ticket.
 3. O utilizador recebe uma notificação quando obtiver resposta.
- **Pós-condições:** O utilizador tem de esperar por uma resposta da equipa.

Blog e Conteúdos

22. Gerir Conteúdo do Blog

- **Ator Principal:** Administrador
- **Descrição:** O administrador pode adicionar ou editar conteúdo no blog, criando e gerindo *posts* relacionados com jardinagem e cultivo.
- **Pré-condições:** O administrador está autenticado.
- **Fluxo Principal:**
 1. O administrador acede à secção de gestão de blog.
 2. O administrador cria ou edita *posts*, inserindo textos, imagens e *tags*.
 3. O sistema guarda e publica o *post*.
- **Pós-condições:** O *post* é publicado ou atualizado no blog.

23. Comentar artigo do Blog

Ator Principal: Utilizador

- **Descrição:** O utilizador pode criar um comentário num blog.
- **Pré-condições:** O utilizador tem de ter o log in feito e tem que haver um blog publicado.
- **Fluxo Principal:**
 1. O Utilizador seleciona o blog que deseja comentar.
 2. O Utilizador fez o comentário no blog.
- **Pós-condições:** O comentário é mostrado para todos a menos que seja ocultado por um ator Administrador.

24. Visualizar Artigos do Blog

Ator Principal: Utilizador

Descrição:

O utilizador pode aceder e visualizar os artigos disponíveis no blog da plataforma.

Pré-condições:

- O blog está configurado e contém artigos publicados.
- O utilizador está autenticado ou acede a área pública do blog (dependendo da política de acesso da plataforma).

Fluxo Principal:

1. O utilizador navega até à secção de blog na plataforma (via menu principal ou link direto).
2. O sistema exibe uma lista dos artigos disponíveis, ordenados por data de publicação ou outra métrica (Ex.: relevância, popularidade).
3. O utilizador seleciona um artigo da lista.
4. O sistema exibe o conteúdo completo do artigo, incluindo texto, imagens e outros recursos multimédia relacionados.

Pós-condições:

- O utilizador visualizou o conteúdo do artigo selecionado.

Gestão de Dados e Relatórios

25. Gerir Categorias

- **Ator Principal:** Administrador
- **Descrição:** O administrador pode criar, editar ou remover categorias de produtos na loja online para organizar o inventário de forma eficiente.
- **Pré-condições:** O administrador deve estar autenticado.
- **Fluxo Principal:**
 1. O administrador acede à secção de "Gestão de Categorias".
 2. O administrador pode adicionar uma nova categoria, editar o nome ou apagar categorias existentes.

3. O sistema atualiza a lista de categorias.

- **Fluxo Alternativo:** O administrador tenta remover uma categoria que está associada a produtos. O sistema exibe uma mensagem de erro.
- **Pós-condições:** As categorias de produtos são criadas, editadas ou removidas de acordo com as ações do administrador.

26. Gerir Produtos

- **Ator Principal:** Administrador
- **Descrição:** O administrador pode adicionar novos produtos à loja, atualizar detalhes de produtos existentes e verificar o stock.
- **Pré-condições:** O administrador está autenticado.
- **Fluxo Principal:**
 1. O administrador acede à secção de gestão de stock.
 2. O administrador verifica ou edita informações de stock de um produto.
 3. O sistema atualiza o stock de acordo com as alterações.
- **Fluxo Alternativo:** O produto não existe no sistema. O administrador recebe uma mensagem de erro.
- **Pós-condições:** O stock é atualizado e os produtos são geridos corretamente.

27. Gerir Localizações

Ator Principal: Administrador

Descrição: O administrador pode criar, alterar e eliminar localizações onde as plantas podem ser colocadas pelos utilizadores, definindo o nome da localização.

Pré-condições:

- O administrador está autenticado na plataforma.
- O sistema possui uma interface para gestão de localizações.

Fluxo Principal:

1. O administrador acede à secção de gestão de localizações no painel administrativo.
2. O administrador pode realizar as seguintes ações:

- **Criar uma localização:**
 - Insere o nome da localização
 - Guarda a nova localização.
- **Alterar uma localização existente:**
 - Seleciona uma localização da lista.
 - Atualiza os dados, como nome, descrição ou condições.
 - Guarda as alterações realizadas.
- **Eliminar uma localização existente:**
 - Seleciona uma localização da lista.
 - Confirma a exclusão.

3. O sistema valida e salva as alterações realizadas pelo administrador.

Fluxos Alternativos:

- **Erro ao salvar dados:**
 - O sistema exibe uma mensagem de erro informando a falha (Ex.: nome da localização já existente, campos obrigatórios não preenchidos) e orienta o administrador a corrigir os dados.

28. Visualizar histórico de tickets

- **Ator Principal:** Utilizador
- **Descrição:** O Utilizador pode visualizar todo o histórico de tickets criado pelo mesmo. Assim sabendo quais é que se encontram com resposta ou fechados.
- **Pré-condições:** O Utilizador tem de ter uma conta e estar com o Log in feito.
- **Fluxo Principal:**
 1. O Utilizador acede à secção do seu perfil.
 2. O Utilizador dentro da secção do seu perfil seleciona histórico de tickets.
 3. O Utilizador pode visualizar todo o seu histórico de tickets.

29. Gerar Relatório de Estatísticas

- **Ator Principal:** Administrador

- **Descrição:** O administrador pode gerar relatórios sobre as estatísticas de vendas, utilizadores e outros dados da plataforma.
- **Pré-condições:** O administrador está autenticado.
- **Fluxo Principal:**
 1. O administrador seleciona o tipo de relatório desejado (vendas, utilizadores, etc.).
 2. O sistema gera o relatório com base nos dados recolhidos.
 3. O administrador visualiza ou exporta o relatório.
- **Pós-condições:** O relatório é gerado e entregue ao administrador para análise.

Notificações e Alertas

30. Receber Notificações e Alertas Administrativos

Ator Principal: Administrador

Descrição: O administrador recebe notificações sobre produtos com stock baixo e alertas quando há um novo ticket de suporte a ser tratado.

Pré-condições: O administrador está autenticado.

O sistema de notificação está ativo e configurado corretamente.

Fluxo Principal:

1. O administrador acede ao painel de administração.
2. O sistema verifica se threshold de stock definido foi ultrapassado e a existência de novos tickets de suporte.
3. Caso algum produto tenha stock abaixo do threshold definido, o sistema envia uma notificação ao administrador no painel e/ou por e-mail.
4. Caso um novo ticket de suporte seja aberto, o sistema envia um alerta ao administrador no painel e/ou por e-mail.
5. O administrador pode clicar na notificação para ser redirecionado à secção relevante (gestão de stock ou suporte ao cliente).

Pós-condições: O administrador foi informado sobre produtos com stock baixo e sobre novos tickets de suporte.

As notificações são registadas no sistema para consulta futura.

31. Receber Notificação Newsletter

Ator Principal: Utilizador

Descrição: O utilizador recebe notificações sobre novos conteúdos na plataforma.

Pré-condições:

- O utilizador está autenticado na plataforma.
- O utilizador ativou a opção de receber notificações da newsletter.
- Há novos conteúdos disponíveis.

Fluxo Principal:

1. O utilizador acede à sua página inicial ou ao painel principal da plataforma.
2. O sistema verifica se o utilizador tem a opção de newsletter ativa.
3. Caso a opção esteja ativa e existam notificações disponíveis, o sistema exibe um alerta visual na interface.
4. O utilizador clica na notificação para visualizar os detalhes do novo conteúdo ou promoção.
5. O sistema redireciona o utilizador para a secção correspondente (Ex.: página do artigo).

Fluxos Alternativos:

- **Newsletter desativada:**
 - O sistema não exibe notificações relacionadas à newsletter para este utilizador.
- **Sem novos conteúdos:**
 - O sistema informa que não há notificações disponíveis.

Pós-condições:

- O utilizador visualizou ou foi notificado sobre o novo conteúdo ou promoção da newsletter.
- As notificações podem ser marcadas como lidas pelo utilizador.

Sensores

32. Enviar Notificações de Alerta para o Sistema

- **Ator:** Sensores
- **Descrição:** O sistema envia notificações de alerta para o utilizador sempre que os sensores detetarem condições adversas nas hortas (como níveis baixos de humidade, temperatura inadequada ou falta de luz).

- **Pré-condições:** Os sensores IoT estão instalados e monitorizando as condições da horta em tempo real.
- **Fluxo Principal:**
 1. Os sensores captam dados em tempo real (ex.: humidade do solo, temperatura do ar).
 2. O sistema identifica uma condição crítica (ex.: humidade abaixo do nível ideal).
 3. O sistema envia uma notificação de alerta ao utilizador através da plataforma web ou aplicação móvel.
- **Pós-condições:** O utilizador é notificado e pode agir para corrigir o problema, como ajustar a irrigação ou proteger as plantas.

33. Monitorizar Condições da Planta

- **Ator:** Sensores
- **Descrição:** O utilizador pode monitorizar as condições de crescimento das plantas, como humidade do solo, temperatura ambiente e luz disponível, através de uma interface na plataforma web.
- **Pré-condições:** Os sensores IoT estão em operação e a plataforma está disponível para o utilizador.
- **Fluxo Principal:**
 1. Os sensores IoT recolhem dados das condições ambientais e do solo ao redor das plantas.
 2. O sistema processa os dados e atualiza a dashboard na plataforma web em tempo real.
 3. O utilizador acede à plataforma para visualizar o estado atual das condições de cultivo.
- **Pós-condições:** O utilizador tem acesso às informações sobre as condições da horta, permitindo que faça ajustes para otimizar o crescimento das plantas.

Outros Casos de Utilização

34. Participar no Quiz Personalizado

- **Ator Principal:** Utilizador
- **Descrição:** O utilizador participa num quiz para conhecer a melhor planta para si, com base no seu estilo de vida e preferências.
- **Pré-condições:** O utilizador tem uma conta registada.
- **Fluxo Principal:**
 1. O utilizador responde a perguntas no quiz.
 2. O sistema analisa as respostas.
 3. O sistema apresenta sugestões de plantas ideais para o utilizador.
- **Pós-condições:** O utilizador recebe sugestões personalizadas de plantas

Anexo D

[Teste Lighthouse Mobile](#)

[Teste Lighthouse Desktop](#)

[Output do VSCode Testes Unitários Pest](#)

[Testes Unitários Pest Realizados](#)

[Testes Angular Realizados](#)

Anexo E

Wireframes

Anexo F

[Workspace Figma](#)

Anexo G

Diagramas MVC

Anexo H

[Diagrama de arquitetura de informação simples](#)

[Diagrama de arquitetura de informação de dashboard de admin](#)

[Diagrama de arquitetura da informação dashboard de utilizador](#)

Anexo I

[CultivUA – Documentação da API](#)

Anexo J

Script DDL SQL – Base de Dados

Anexo K

[Imagens e Vídeos Demonstrativos do Website](#)

Anexo L

[Mockups Manual de Instalação e Cartão com Código do Kit](#)